BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

SEARCH IN SPEECH RECORDINGS BASED ON SEMANTIC VECTORS

HLEDÁNÍ INFORMACÍ V NAHRÁVKÁCH ŘEČI POMOCÍ SÉMANTICKÝCH VEKTORŮ

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR **Bc. DOMINIK BOBOŠ**
AUTOR PRÁCE

SUPERVISOR **Ing. PETR SCHWARZ, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2024

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Master's Thesis Assignment

156970

Institut:	Department of Computer Graphics and Multimedia (DCGM)
Student:	**Boboš Dominik, Bc.**
Programme:	Information Technology and Artificial Intelligence
Specialization:	Sound, Speech and Natural Language Processing
Title:	**Search in speech recordings based on semantic vectors**
Category:	Speech and Natural Language Processing
Academic year:	2023/24

Assignment:

1. Learn about neural network-based techniques for converting text into semantic vectors and using these vectors for information retrieval.
2. Study the techniques that obtain compatible semantic vectors from text and audio (joint representation models)
3. Prepare a dataset for studying audio search algorithms using semantic vectors from a provided database of audio recordings with transcripts. The searched pattern (word, phrase) can be entered by text or audio (query by example).
4. Perform information retrieval experiments using semantic vectors in recordings, choose an appropriate evaluation metric, and evaluate the experiments.
5. Discuss the obtained results from the point of practical use of the method for search in speech recordings.

Literature:
According to the consultation with the supervisor.

Requirements for the semestral defence:
Points 1 to 3

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor:	**Schwarz Petr, Ing., Ph.D.**
Head of Department:	Černocký Jan, prof. Dr. Ing.
Beginning of work:	1.11.2023
Submission deadline:	17.5.2024
Approval date:	9.11.2023

Abstract

In the current era of information overload, efficient methods for information retrieval are crucial. This thesis summarises methods for obtaining vector representations for text and audio, also known as semantic vectors. We took a deeper look at joint-representation models such as SpeechT5 and SeamlessM4T, which transform these various forms of input into one shared vector space. Based on these models, we built a system which allows us to search in data regardless of the modality. In order to evaluate the proposed solution on semantic search tasks, apart from standard keyword spotting tasks, we labelled a dataset to capture similar semantic meanings of the keywords or phrases. Finally, we conducted several experiments, where we explored the possibilities of the models used by limiting the context seen during finetuning or involving text-to-speech (TTS) systems to improve overall performance.

Abstrakt

V současné době přetížené informacemi jsou efektivní metody vyhledávání informací velice žádané. Tato práce shrnuje metody pro získávání vektorových reprezentací pro text a zvuk, známé také jako sémantické vektory. Podívali jsme se hlouběji na multimodální modely, jako jsou SpeechT5 a SeamlessM4T, které transformují tyto typy vstupu do jednoho sdíleného vektorového prostoru. Na základě těchto modelů jsme vybudovali systém, který nám umožňuje vyhledávat v datech bez ohledu na modalitu. Abychom mohli vyhodnotit navrhované řešení, kromě standardního rozpoznávání klíčových slov, také pro úlohy sémantického vyhledávání, manuálně jsme označili datovou sadu pro zachycení podobných sémantických významů klíčových slov nebo frází. Nakonec jsme provedli několik experimentů, kde jsme prozkoumali možnosti modelů omezením pozorovaného kontextu během dotrénovaní neuronové sítě nebo zapojením systémů převodu textu na řeč (TTS) ke zlepšení celkového výkonu.

Keywords

shared embedding space, semantic vectors, audio embeddings, word vectors, transformers, SSL models, Joint-representation models, multimodal models, keyword spotting, semantic search, information retrieval

Klíčová slova

sdílený vektorový prostor, sémantické vektory, vektorová reprezentace audia, slovní vektory, transformery, SSL modely, multimodální modely, detekce klíčových slov, sémantické prohledávání, vytěžování informací

Reference

BOBOŠ, Dominik. *Search in speech recordings based on semantic vectors.* Brno, 2024. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Petr Schwarz, Ph.D.

Rozšířený abstrakt

V súčasnej dobe preťaženej informáciami a obrovským tokom dát je nevyhnutné zamerať sa len na tie skutočne relevantné informácie. Nájdenie tých správnych údajov vyžaduje rýchle a spoľahlivé systémy na prehľadávanie v obrovskom objeme dát. Takéto vyhľadávacie mechanizmy sú kľúčové pre mnohé profesie, ako sú orgány činné v trestnom konaní, tajné služby, investigatívni novinári či bežné profesie pracujúce s informáciami.

V priebehu rokov sa preto vyvinuli techniky ako vzhľadávanie kľúčových v audiu (KWS) či prepis reči do textu (STT). V KWS systémoch môžeme kľúčové slovo zadať buď zvukovou nahrávkou – Query-by-Example (QbE) alebo pokročilé systémy KWS podporujú aj textové vstupy – Query-by-text. STT nám umožňuje po prepise používať rovnaké techniky ako v prípade textu, ale STT vytvára významné množstvo chýb. Avšak tieto techniky nijako nezachytávajú aj kontext a význam jednotlivých slov či viet.

Cieľom tejto práce bolo riešiť všetky vyššie uvedené problémy do jedného spoločného systému. Preto bolo nevyhnutné získať takú reprezentáciu textu a reči, ktorá odráža aj kontext – takzvané sémantické vektory. V práci boli zhrnuté jednotlivé prístupy získanie vektorovej reprezentácie pre text a audio. Hlavne boli opisané modely využívajúce architektúru "transformers", pre text to boli hlavne BERT a LaBSE a pre audio Wav2vec2 a HuBERT. Keďže chceme vytvoriť unifikovaný systém, preskúmali sme aj multimodálne modely spoločnej vektorovej reprezentácie a to modely SpeechT5 a SeamlessM4T.

Tie nám umožnili nezávisle od typu vstupu používať zvukové alebo textové dáta. V tejto práci sa posudzovala preveditelnosť a použiteľnosť takéhoto prístupu a ich pripravenosť nahradiť aktuálne používané systémy.

S týmito multimodálnymi modelmi sme navrhli vyhľadávací algoritmus, kde sme sa snažili využiť potenciál predtrénovaných modelov SpeechT5. Skúmalo sa, ktoré nastavenie funguje najlepšie a to bolo postavné ako základný systém pre ďalší vývoj. V procese vývoja sme objavili viaceré nastavenia, ktoré ovplyvňovali kvalitu výsledkov, ako napríklad správne narovnanie prehľadávacieho okna pri porovnávaní naskrz modalít.

Taktiež sme vykonali niekoľko experimentov na zlepšenie základného systému. To zahŕňalo dotrénovanie modelu SpeechT5, pri ktorom sme sa sústredili na obmedzenie viditeľného kontextu a analyzovali sme jeho vplyv na presnosť. Experimentovali sme s novou, robustnejšou architektúrou modelov pre spoločnú reprezentáciu – SeamlessM4T, a pokúsili sme sa prispôsobiť vlastnosti daného modelu na vytvorený vyhľadávací algoritmus. Nakoniec sme zvýšili presnosť navrhovaného systému využitím mechanizmu hlasovej syntézy – text-to-speech (TTS), ktorý sa zaintegroval ako metóda na predspracovanie textového vstupu. Tento prístup sa javil ako ten s najvyšou presnosťou a stabilitou

Aby sme mohli vyhodnotiť presnosť systémov pre úlohy sémantického vyhľadávania, manuálne sme vyznačili pomocou synonymického slovníka sémanticky podobné slová pre úlohu sémantického vyhľadávania kľúčových slov. A taktiež sme okrem toho označili aj významovo blízke vety pre úlohu vyhľadávania sémantických fráz. Systémy boli vyhodnotené aj na úlohu klasického vyhľadávania kľučových slov a najlepšie KWS systémy boli porovnané aj s komerčným riešením spoločnosti Phonexia.

Získané výsledky ukazujú veľký potenciál tohto nového prístupu pre vyhľadávanie v reči a texte. Avšak pre komerčné nasadenie je potrebný ďalší vývoj. Nasledujúci vývoj primárne zahŕňa zlepšenie presnosti, zníženie miery chýb pre falošné označenia a hlavne zvýšenie rýchlosti extrakcie vektorovej reprezentácie, a aj rýchlosť samotného vyhľadávacieho procesu.

Search in speech recordings based on semantic vectors

Declaration

I hereby declare that this Master's thesis was prepared as an original work by the author under the supervision of Ing. Petr Schwarz, Ph.D. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....
Dominik Boboš
May 14, 2024

Acknowledgements

I would like to thank my supervisor Ing. Petr Schwarz, Ph.D for his guidance, willingness, patience and valuable suggestions. I would also like to thank members of BUT Speech@FIT group and my colleagues at Phonexia for sharing their know-how, which helped me throughout my work. Furthermore, I would also like to thank my family for their endless support. This work was also supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254).

Contents

Chapter 1

Introduction

In the current era of information overload, it is essential to deal only with the relevant information. Finding the right pieces of data requires fast and reliable information retrieval systems. Such search mechanisms are crucial for many professions, namely law-enforcement agencies, secret services, journalists, and common professions that work with information.

The simplest system is to search for keywords within text documents. This may be sufficient for many use cases, however, it also carries many drawbacks. The query needs to match the case exactly, slight changes in spelling or different word forms will ruin the quality of the approach.

In addition, information flows through more sources than just in text form. The most natural way for human communication is speech. Processing speech signals or audio, in general, brings even more challenges. In speech, not even exact matching is reliably working.

Therefore, techniques like *Keyword Spotting* (KWS) or transcribing *Speech-To-Text* (STT) have evolved over the years. KWS works similarly to search in text. We enter the keyword either by audio – Query-by-Example (QbE) or advanced KWS systems support text queries – Query-by-text. STT allows us to use the same techniques after transcription as in the form of text, but STT creates a significant number of errors and is not accessible for low-resource languages.

Additionally, it is often desirable to capture synonyms, other word forms, and similar contexts. We either do not remember the exact keyword or want to catch semantically close terms.

This thesis aims to address all of the issues above into one solution. Thus, it is necessary to obtain a representation of text and speech that reflects the context. The goal of this work is to apply current semantic vector representation approaches to text and speech and try to search within these vectors. For the most efficient use, these modalities should be interconnected, and thus we will examine joint-representation models. This allows us to use audio or text queries interchangeably. This thesis objective is to evaluate the feasibility and usability of this approach and determine whether it can replace the currently deployed systems.

The thesis is organised as follows. In Chapter 2 and 3, we will provide an overview of techniques to obtain text and audio vector representations. In Chapter 4, we will examine architectures of joint-representation models. In Chapter 5, we will describe the datasets we are using in development and Chapter 6, we present evaluation metrics. Chapter 7 presents the proposed search algorithm and the initial system. Finally, Chapter 8 describes experiments and the results with the joint-representation models.

Chapter 2

Vector representation for text

To process a large amount of text from the real world in the world of binary digits, it is required to properly encode or convert the desired input. To search exact phrases – keywords, the plain text form is sufficient. However, even for simpler tasks some level of fuzziness is welcomed and expected. Thence the main objective is to evaluate various properties of each word to understand the context and therefore bring more satisfactory search results.

2.1 Text representation evolution

The earliest language representations tried to evaluate various properties of each word manually. One of the approaches was *Semantic Differential* [34] which evaluates properties for words such as opposites (sweet-bitter, warm-cold, etc.) or other properties and fills this data into a vector. The main drawback of the Semantic Differential is that it captures only predefined attributes and in addition, it requires a manual annotation which is subjective, exhaustive and expensive. The more naive approach was *One-hot encoding*, which takes the idea of a vector of vocabulary size and sets "1" for the position of the word otherwise "0" is set.

This idea is extended to documents, bringing us to another approach to text representation called *Bag of Words* (BoW). BoW represents a document as a collection of words. It takes a vocabulary-sized vector, where a "1" is added to the corresponding index each time a word is present in a sentence, and "0" is placed elsewhere [19]. The difference between One-hot encoding and BoW is that One-hot encoding provides binary vectors indicating the presence or absence of words, and BoW provides counts of word occurrences in a sentence.

A more sophisticated approach is *term frequency-inverse document frequency* (TF-IDF), which stems from BoW and takes into account the frequency of each word in the document and the frequency of the word in the entire corpus of documents. TF-IDF $tf - idf(t, d)$ of term t, of document d is given by the following formula:

$$\text{tf-idf}(t, d) = tf(t, d) \times idf(t, D)\,. \tag{2.1}$$

Term frequency $tf(t, d)$ of term t in document d is given by:

$$\text{tf}(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}\,, \tag{2.2}$$

where $f_{t,d}$ is the number of times that term t occurs in document d, $\Sigma_{t' \in d} f_{t',d}$ represents the total count of terms in document d. Inverse document frequency (IDF) $idf(t, D)$ of term t, where D represents all documents is a measure of the amount of information conveyed by a word which is given by the following equation:

$$\text{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}, \tag{2.3}$$

where N is the number of all documents in the corpus and $|\{d \in D : t \in d\}|$ represents the number of documents in the corpus containing term t. TF tells how important the term is in the document and IDF shows how unique the term is in general – frequent terms are usually less important. These methods have been widely used in *Natural Language Processing* (NLP) for tasks such as text classification, information retrieval or a weighting factor [40].

2.2 Unsupervised techniques for word embeddings

However, the methods described in section 2.1 have limitations: `i)` lack of capturing the meaning of words or the relationships between them, and `ii)` high-memory demands. As the words in the mentioned methods are represented as a vector of vocabulary size $\mathbf{v}$, all words are represented by matrix $\mathbf{M}$ of size $\mathbf{v} \times \mathbf{v}$. Therefore it is unsuitable for large dictionaries as $\mathbf{M}$ is very sparse.

To address the issue `ii)` to decrease the dimensions of matrix $\mathbf{M}$, respectively dimensions of a vector $\mathbf{v}$, researchers involve tokenisation by *Byte-pair encoding* on word pieces. To solve problem `i)`, they proposed *semantic vectors*, which represent words as vectors in a high-dimensional space, where the distance between vectors reflects the semantic similarity between words. One such model is *Word2vec* introduced by Tomáš Mikolov [29].

2.2.1 Word2vec

Word2vec is a neural network-based technique for generating word embeddings, which are vector representations of words in a high-dimensional space. Word2vec is trained on a large corpus of text data and learns to predict the context of a word based on its surrounding words. The resulting word embeddings can be used for various natural language processing tasks such as text classification, sentiment analysis, and machine translation [30]. The model proved to solve analogies by using vector arithmetic on the word embeddings. For example, to find the missing word in the analogy "Man is to king as woman is to ?", the vector difference can be computed between "man" and "king", add it to the vector of "woman", and find the closest word vector to the result. This would return "queen" as the answer. Mikolov introduced two methods for training Word2vec: `i)` *Continuous Bag of Words* (CBOW) model and `ii)` The Skip-gram model [29].

`i)` CBOW is based on the idea that the meaning of a word can be inferred from the words that surround it in a sentence. CBOW uses a neural network to learn word embeddings by predicting a target word given its context words. For example, given the sentence "He is a great jazz musician.", CBOW would try to predict the word w_t = *"great"* based on the words $w_{\{t-n, t-1, t+1, t+n\}}$ = {*"is"*, *"a"*, *"jazz"*, *"musician"*} for a window size $ws = 2 \times n = 4$. Thus probability $p\left(w_t \mid w_{t-n}, w_{t-1}, w_{t+1}, w_{t+n}\right)$ should be for w_t = *"great"* higher than

for w_t = *"glasses"*. The objective function J_θ of all terms T with a window size $2 \times n$ is given by the following equation [30]:

$$J_\theta = \frac{1}{T} \sum_{t=1}^{T} \log p\left(w_t \mid w_{t-n}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+n}\right) , \tag{2.4}$$

where w_t is the word to predict and $w_{\langle t-n,t+n \rangle}$ are words at the input representing the context of the word w_t.

During the training of the neural network, the objective function of CBOW is to maximise average log probability from equation 2.4.

ii) The Skip-gram model predicts the context words given a target word. Thus it is the opposite of the CBOW. For the given sentence "He is a great jazz musician.", for the word $w_t = great$ Skip-gram predicts the surrounding words. For a window size of $ws = 2 \times n = 4$ it should predict words $w_{\{t-n,t-1,t+1,t+n\}} = \{$*"is"*, *"a"*, *"jazz"*, *"musician"*$\}$. The objective function K_θ of all terms T with a window size $2 \times n$ is given by the following equation [30]:

$$K_\theta = \frac{1}{T} \sum_{t=1}^{T} \sum_{-n \leq j \leq n, \neq 0} \log p\left(w_{t+j} \mid w_t\right) , \tag{2.5}$$

where w_t is the input word and w_{t+j} are words to predict in the context of the word w_t.

A simple neural network without a non-linear hidden layer is used for both models. The projection layer, which has a log-linear classifier, is shared for all words. Therefore, all words are projected to the same position by averaging their vectors [29]. The architecture of the models is shown in Figure 2.1.

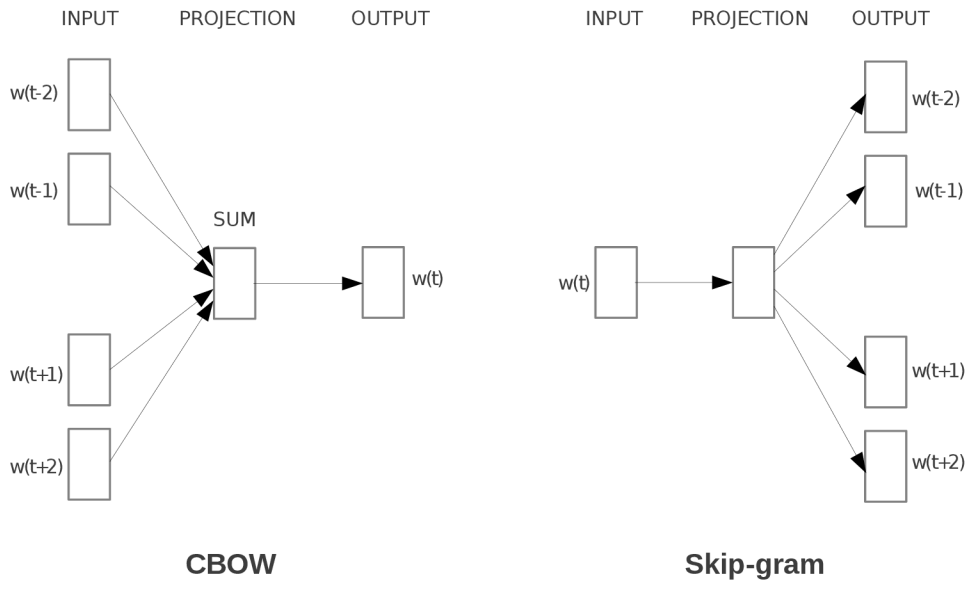


Figure 2.1: Word2vec – CBOW and Skip-Gram architectures. Taken from [29]

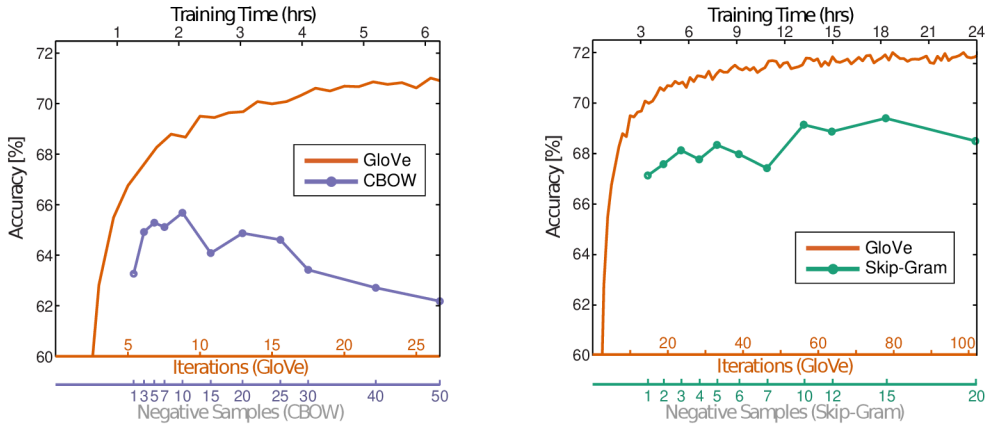
2.2.2 GloVe

Another unsupervised learning algorithm for the representation of words as vectors is from Stanford University by J.Pennington – *Global Vectors for Word Representation* (GloVe) [36]. GloVe aims to solve drawbacks in methods like Latent Semantic Analysis (LSA) [11] or Skip-Gram Word2vec. GloVe improves LSA's performance on the analogy task and unlike Word2vec, GloVe makes full use of the corpus statistics.

GloVe does not rely on local context windows. It is trained on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. The main intuition behind the model is that ratios of word-word co-occurrence probabilities have the potential for encoding some form of meaning. Let $\mathbf{X}$ be the matrix of word-word co-occurrence counts, where elements $\mathbf{X}_{ij}$ hold the number of times word j appears in the context of word i. Also, let $X_i = \sum_k X_{ik}$ be the total number of words that appear in the context of word i. Then, $P_{ij} = P(j \mid i) = X_{ij}/X_i$ is the probability of word j appearing in the context of word i. The model's objective is to minimise the squared errors of the logarithms of the probabilities, using a bilinear function to estimate them. The objective function of GloVe is defined as follows:

$$J = \sum_{i,j=1}^{V} f(P_{ij})(\mathbf{w_i^T}\tilde{\mathbf{w}}_\mathbf{j} + \mathbf{b_i} + \tilde{\mathbf{b}}_\mathbf{j} - \log P_{ij})^2 \,, \tag{2.6}$$

where $\mathbf{w_i}$ and $\tilde{\mathbf{w}}_\mathbf{j}$ are the word vectors for the i-th and j-th words, $\mathbf{b_i}$ and $\tilde{\mathbf{b}}_\mathbf{j}$ are their respective biases, P_{ij} is the probability of word j appearing in the context of word i, and $f(x)$ is a weighting function that assigns less weight to the more frequent word pairs. Pennington claims that GloVe outperforms CBOW and Skip-gram Word2vec on the word analogy task, as can be seen in Figure 2.2. The model is scalable and achieved state-of-the-art performance on word analogy, word similarity, and named entity recognition tasks [36].



(a) GloVe vs CBOW

(b) GloVe vs Skip-Gram

Figure 2.2: Comparison of the accuracy between Word2vec [30] and GloVe [36] on the word analogy task as a function of training time, which is determined by the number of iterations for GloVe and by the number of negative samples for `(a)` CBOW and `(b)` Skip-gram. (300-dimensional vectors were trained on the 6B token corpus – Wikipedia 2014 + Gigaword 5) [36].

2.3 Language representation models

Language representation models aim to capture the meaning of natural language in a way that can be used for various downstream tasks, such as sentiment analysis, machine translation, and question-answering. These models learn to represent words and sentences as vectors in a high-dimensional space – *semantic vectors*, where similar words and sentences are close together and the opposite ones are far apart [43]. One of the most important types of language representation models is the Transformers [44], which is based on attention mechanisms.

With the rise of Transformers, many derivatives of the architecture evolved. One of them is *Bidirectional Encoder Representations from Transformers* (BERT) introduced by Google in 2018 [24], which happened to be a baseline for further models [43].

2.3.1 BERT

BERT is a multi-layer bidirectional Transformer encoder for language representation and many NLP tasks. It is pretrained on the BookCorpus and English Wikipedia. BERT shows the importance of bidirectional pretraining as opposed to a unidirectional approach like in [37] and presents unsupervised pretraining by using two tasks: `ii)` Masked LM and `ii)` Next Sentence Prediction (NSP) [24].

`i)` Masked LM is based on masking tokens at the input. To train a deep bi-directional representation model BERT authors masked 15 % of all tokens from the corpus.

Out of these masked 15 % of the input sequences, 80 % has a randomly chosen n-th token of the sequence, which is masked using `[MASK]` token. For example, from the sentence "`[CLS]` I have a new computer `[SEP]`" a randomly chosen 5-th token will be masked – "`[CLS]` I have a new `[MASK]` `[SEP]`", where `[CLS]` is a special token added in front of every input example and `[SEP]` is a special separator token. Afterwards, the masked token is predicted.

Next 10 % of the masked input sequences has the n-th token replaced by a random word – "`[CLS]` I have a new **`water`** `[SEP]`".

The rest 10 % of the times, the chosen token remains unchanged. Which should put bias to the representation model to prefer the actual observed word. The purpose of this method is that the Transformer encoder cannot predict which tokens it will need to generate or which ones have been randomly substituted by other words. Therefore, it has to maintain a contextual representation of every input token based on its distribution [24].

`ii)` Next Sentence Prediction (NSP) Language modelling is typically a word-level representation, but it does not explicitly model the semantic relation between two sentences. However, this relation is crucial for many NLP tasks, such as question answering or natural language inference, that require reasoning and inference skills.

To learn relations between sentences BERT is pretrained for a binarised NSP task. First, sentence pairs `A` and `B` for training are chosen, where `A` is the currently processed sequence. 50 % of the times `B` is the actual next sentence from the corpus and is labelled with flag `IsNext`. The other 50 % times sentence `B` is not the next sentence but a randomly chosen sequence from the corpus and it is labelled as `NotNext`. Authors claim that NSP is very beneficial to question answering, and after proper finetuning, for natural language inference too [24].

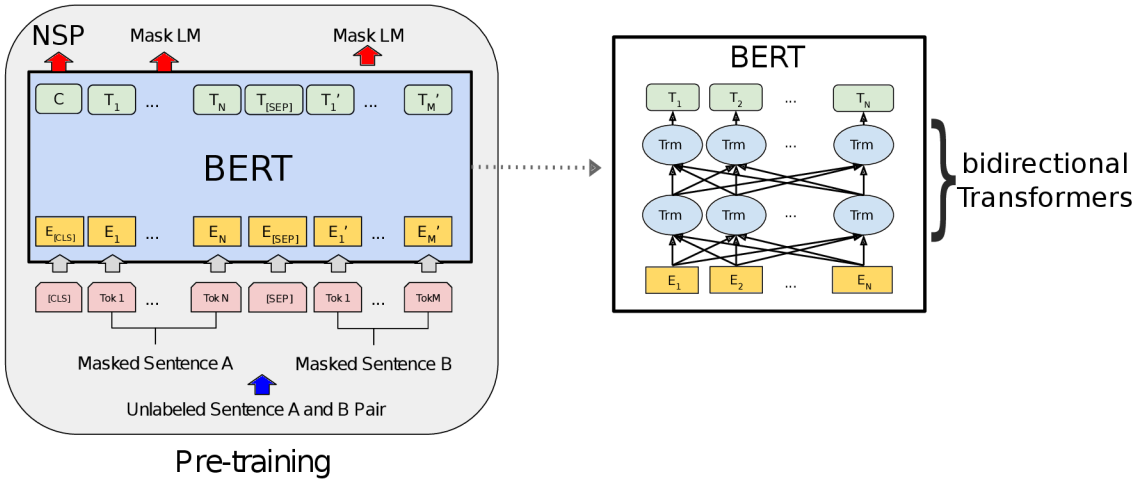


Figure 2.3: BERT pretraining architecture (finetuning architecture is analogical). BERT takes sentence pairs with masked tokens as input. C is the vector of tokens representing the prediction of the next sentence for binarized NSP. Adapted from [24].

The architecture of BERT pretraining is shown in Figure 2.3. The self-attention mechanism in the Transformer makes BERT versatile enough to handle many downstream tasks, which simplifies the finetuning process. For each task, it is enough to change input pair sentences `A` and `B` to the desired input. a model of such quality is necessary for proper word representation to get semantic vectors.

2.3.2 LaBSE

Unfortunately, BERT is not suitable as a cross-lingual model, as it is trained on English corpora only. The multilingual model version of BERT – mBERT is pretrained from monolingual corpora in 104 languages. However, mBERT is not equally effective for all languages [38] – *Language-agnostic BERT Sentence Embedding* (LaBSE) [16] comes in place to solve this language inequality.

LaBSE is a multilingual BERT model designed to generate language-agnostic sentence embeddings for 109 languages. The pretraining process of LaBSE combines masked language modelling (MLM) with translation language modelling (TLM). MLM works the same way as in the BERT model – MLM is explained in detail in subsection 2.3.1.

TLM is an extension of MLM, where instead of BERT's pairs of word streams, TLM uses pairs of parallel sentences in two different languages – *lang 1*, *lang 2* [10]. An additional difference is that it masks words in both the source and target sentences in a complementary manner. For example, let *lang 1* to be English and *lang 2* to be German. Source sentence in English is "*He is very tired and exhausted.*" and the corresponding target sentence in German "*Er ist sehr müde und erschöpft.*". The source and target sentences can be masked the following way – "*He is* `[MASK]` *tired and* `[MASK]`." and "*Er* `[MASK]` *sehr* `[MASK]` *und erschöpft.*". Thus, to predict a masked word in *lang 1*, the model can either attend to surrounding words of *lang 1* or take the word from *lang 2*. This should align the representation of both languages. The position is reset each time to the beginning to help the alignment process [10].

LaBSE adapts dual-encoder architecture by M.Guo [18] as it is an effective approach for cross-lingual training. The dual-encoder architecture encodes two sequences x and y in parallel with separate encoders. Then, the embedding space similarity $\phi(x, y)$ is given by a dot-product of the encoded input to vectors $\mathbf{u}, \mathbf{v}$: $\phi(x, y) = \mathbf{u}^T \cdot \mathbf{v}$ [18], the architecture is shown in Figure 2.5.

For LaBSE the source and target sentences are encoded separately. The sentence cross-lingual embeddings are trained with in-batch negative sampling[1] for a translation ranking task [16]. To set a more distinctive and compact boundary between the target translation and nearby non-translations LaBSE is using *Additive Margin Softmax*. It introduces a large margin, m, around positive pairs in the scoring function – it applies as follows: [48]:

$$\phi'(x_i, y_j) = \begin{cases} \phi(x_i, y_j) - m & \text{if } i = j \\ \phi(x_i, y_j) & \text{if } i \neq j \end{cases} \tag{2.7}$$

The visualisation of what Additive Margin Softmax does to an original embedding space is depicted in Figure 2.4.

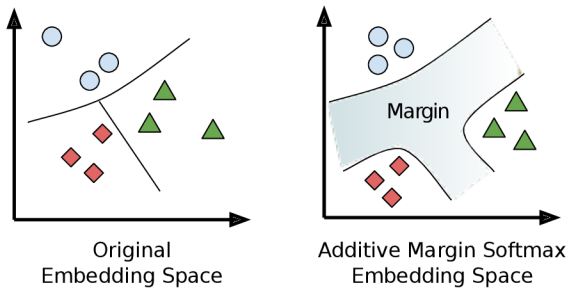
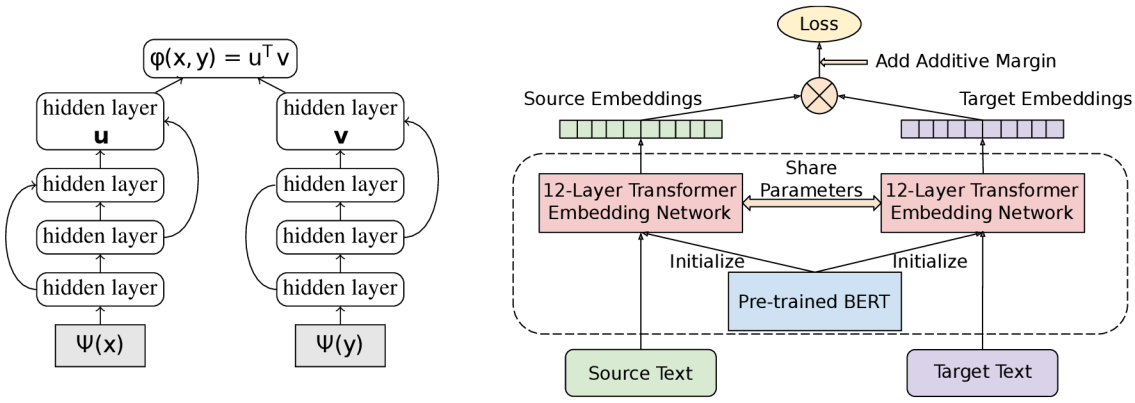


Figure 2.4: A comparison of embedding spaces with and without additive margin softmax. The shapes correspond to sentences and the same shapes are translations in various languages. Taken from [48].

LaBSE shows great performance on large-scale bilingual-text retrieval, and downstream classification tasks and it boosts translation ranking performance [16]. It shows it makes sentence embedding space of good quality which allows to use the semantic vectors in a truly language-agnostic way. The LaBSE architecture is shown in Figure 2.5.

[1]The objective function is to maximise the similarity between the source sentence and the corresponding true translation and minimise it with other incorrect translations.



(a) Dual-Encoder architecture

(b) LaBSE architecture

Figure 2.5: (a) a dual-encoder architecture consists of two groups of hidden layers. One group encodes the source sentence x into a vector, and the other encodes the target sentence y. The score $\phi(x, y)$ is the dot product of these encoded vectors. Taken from [18]. (b) LaBSE architecture – a dual encoder model with BERT initialisation. Taken from [16].

Chapter 3

Vector representation for audio

Vector representation for audio respectively speech is a more challenging task than for text. Speech signals are more diverse than text. For example, the word "sunset" has only one written form, but it is pronounced differently each time due to varying speed, speaker, volume, tone, or environmental noise.

For a long time, the standard approach for speech processing was to extract some features from audio signals and then process them accordingly to the specific task, such as speech recognition, speaker identification, or speech synthesis. Some of the common features that were used are [1]:

1. *Discrete cosine transform* (DCT) – It is a mathematical operation that converts a signal into a sum of cosine functions with different frequencies and amplitudes [49].
2. *Fast Fourier transform* (FFT) – FFT is an efficient algorithm for computing the discrete Fourier transform (DFT), which decomposes a signal into a sum of complex exponential functions with different frequencies and amplitudes, thus it converts the signal from the time domain to the frequency domain.
3. *Mel-frequency cepstral coefficients* (MFCC) – They are based on the human perception of sound and capture the signal's spectral envelope.
4. *Linear predictive coding* (LPC) – It models the vocal tract as a linear filter and estimates its coefficients from the signal.
5. *Linear predictive cepstral coefficients* (LPCC) – LPCC are derived from the LPC coefficients by applying a discrete cosine transform (DCT).

To obtain better quality features or embeddings than the ones based purely on acoustics, these features could be used as input for clustering. GMM-HMM models or neural nets generate different types of features [1]. Clustering is a technique for grouping similar data points based on some distance or similarity measure. GMM-HMM models are a combination of Gaussian mixture models (GMM) and hidden Markov models (HMM) that are used to model some statistical distribution and temporal dynamics of speech signals. Neural-net training is a process of learning some parameters of a neural network. The neural net is a computational model that consists of multiple layers of interconnected units that can perform complex nonlinear transformations of the input. Features or embeddings are some encoded representations of the input data that are extracted or learned by these techniques. These methods can capture the relevant information for the given task [1]. The evolution of signal processing for the past decades is depicted in Figure 3.1.

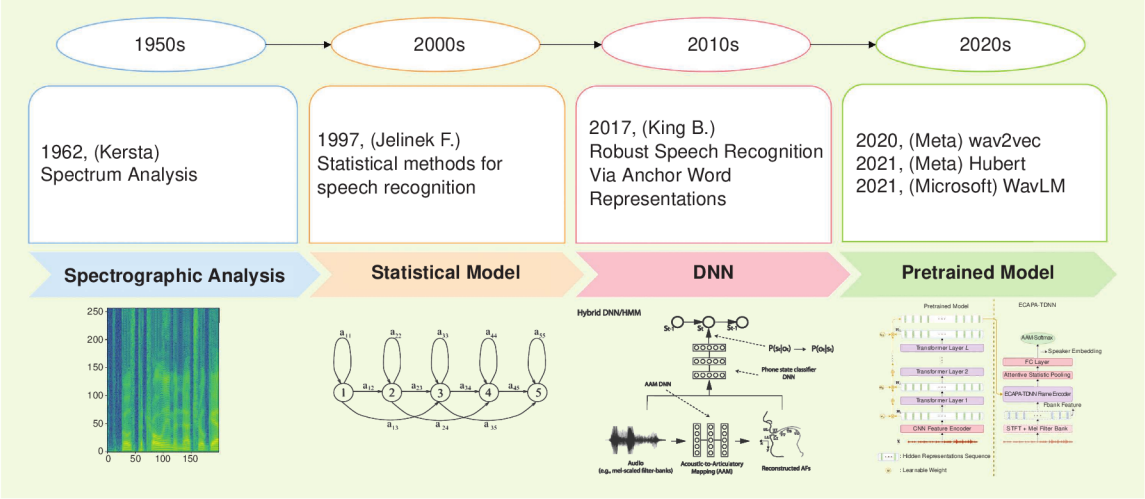


Figure 3.1: Evolution of speech recognition techniques. Adapted from [49].

Lately, research is moving towards using raw audio to learn distinct features directly from waveform [49]. Researchers tried to bring Mikolov's idea of *Word2vec* [29] to audio – Yu-An's *Audio Word2Vec* [6] or Ashwin's *Sound-Word2Vec* [46]. They proved that the concept used in the text has potential for speech as well. *Sound-Word2Vec* works decently with onomatopoeia and Audio Word2Vec shows potential for keyword spotting (KWS) task. However, both of these papers struggle with polysemous words. This idea was taken and reinvented further with *Word2Vec* [42]. The research in unsupervised speech representation has made a huge leap forward in recent years. Architectures like Wav2vec2 [4], HuBERT [21] or WavLM [5] brought improvement for *automatic speech recognition* (ASR) systems.

3.1 Keyword spotting

Keyword Spotting (KWS), also referred to as Spoken Term Detection (STD) is a task with the primary objective of detecting predefined keywords or phrases within continuous speech. Two main approaches can be defined for KWS: `i)` Query-by-Text (QbT) and `ii)` Query-by-Example (QbE).

Query-by-text KWS is an approach when the query is given in the textual form. For QbT KWS, it is assumed that the target language is well-documented, with a lot of resources such as transcribed data, phoneme sets, and pronunciation dictionaries to train a phoneme recogniser. Text queries have to be automatically transcribed into strings of phonemes or other sub-word units. The units can then be converted to the signal form. The differences between the textual representation of queries and the actual spoken form of the search data make the QbT KWS application difficult for low-resource languages [15].

Query-by-example, as opposed to QbT KWS, uses the actual speech signal as a query. QbE KWS systems search for a spoken query within a set of recordings with speech, delivering a list of detections with their scores and timestamps. Therefore, QbE KWS is suitable even for low-resource languages [15].

The following approaches are common in QbE KWS systems:

Dynamic Time Warping (DTW) Dynamic time warping (DTW) is an algorithm used to find the shortest distance and compare two time series data when the time indices are not synchronised [31].

Statistical modelling (HMM/GMM) It is a simplified model of ASR systems where we use only two models – one for the keyword, and one for the background model.

By using Hidden Markov Models (HMM) and Gaussian Mixture Models (GMM) techniques the keyword model is trained on features of the keyword, and the background model is trained on features from all terms in the training dataset [15].

When matching the utterances, respectively features of the input, their likelihood ratios are estimated by both the keyword model and the background model, which yield the scores for the sequence and help us to choose the hypothesis with the higher probability and make the decision [45].

Statistical modelling (HMM/NN) This concept uses Neural Networks (NN) instead of GMM for estimating the likelihood of the samples in GMM states. Thus, instead of fitting GMM by the Expectation–Maximization algorithm, NNs are trained to learn the distribution [45]. NNs help to capture complex patterns and relationships within the data better than GMMs.

Seq2Seq approach With better and more accessible hardware, Seq2Seq approaches have also been explored for the KWS task. One of the approaches is to train the Seq2Seq model to output a special token whenever it encounters the keyword in the input speech. During inference, the presence of this special token in the output sequence indicates that the keyword was spotted in the input [26].

3.2 Wav2vec

Wav2vec presented by Meta in 2019 [42] is a model for unsupervised pretraining for improvement of supervised speech recognition. Wav2vec is a type of *Convolutional Neural Network* (CNN) that processes raw audio waveforms as input and produces a general representation which can then serve as an input for a speech recognition system.

It is based on predicting future samples from a given context in the input waveform. But first the input samples $\mathbf{x}_i$ of the raw audio $\mathcal{X}$ are encoded to a feature representation $f : \mathcal{X} \mapsto \mathcal{Y}$. $\mathcal{Y}$ is a CNN encoder which encodes $30\,ms$ of $16\,kHz$ with a $10\,ms$ stride.

Then *context network* is applied: $g : \mathcal{Y} \mapsto \mathcal{Z}$ which takes gradually n samples from $\mathcal{Y}$: $(\mathbf{y}_i, \ldots, \mathbf{y}_{i-n})$ as a history and creates one tensor $\mathbf{z}_j = g(\mathbf{y}_i, \ldots, \mathbf{y}_{i-n}), z_j \in \mathcal{Z}$.

The modelling of the data distribution is more accurate by using the encoded input $\mathcal{X}$ to $\mathcal{Z}$, since doing it straight from the signal is more challenging [42].

The model is trained by minimising *contrastive loss* objective function. Let k be the count of samples to predict the encoded sequence $\mathcal{Y}$. Then the goal is to correctly identify a sample $\mathbf{y}_{i+k}$ from the future while uniformly choosing "distractor" samples $\tilde{\mathbf{y}}$ from distribution p_n from audio sequence. Where $p_n(\mathbf{y}) = \frac{1}{T}$ and T is the sequence length. For prediction steps of K compute loss $\mathcal{L}$ as follows:

$$\mathcal{L} = \sum_{k=1}^{K} \left(- \sum_{i=1}^{T-k} \left(\log \sigma \left(\mathbf{y}_{i+k}^{\top} h_k \left(\mathbf{z}_i \right) \right) + \underset{\tilde{\mathbf{y}} \sim p_n}{\lambda \mathbb{E}_E} \left[\log \sigma \left(-\tilde{\mathbf{y}}^{\top} h_k \left(\mathbf{z}_i \right) \right) \right] \right) \right), \tag{3.1}$$

where $\sigma(\dots)$ is sigmoid and $\sigma\left(\mathbf{y}_{i+k}^{\top} h_k\left(\mathbf{z}_i\right)\right)$ is the probability of the true prediction of sample $\mathbf{y}_{i+k}$, h_k is step-specific affine transformation, λ is set to the count of $\tilde{\mathbf{y}}$. Wav2vec's ability to carry context in the obtained representation shows great potential for unsupervised boosting of speech-to-text systems [42].

3.3 Wav2vec2

Ideas from *Wav2vec* proved to be efficient in boosting *Word Error Rate* (WER) of speech-to-text systems. *Wav2vec2* [4] extend the proposed solution with updated architecture with the use of Transformers.

Wav2vec2 still relies on the dual model architecture – `(i)` *CNN encoder* and `(ii)` *contextualized representation.*

CNN Encoder is based on several blocks of temporal convolution with a normalisation layer with *Gaussian Error Linear Unit* (GELU)[1] activation function. Again, the raw audio is first encoded $f : \mathcal{X} \mapsto \mathcal{Y}$. Then $\mathcal{Y}$ is discretised with quantisation module $q : \mathcal{Y} \mapsto \mathcal{Q}$ to a finite set of speech representations by using *product quantisation*[2], with given G codebook groups and V entries. The feature encoder output $\mathbf{y} \in \mathcal{Y}$ is mapped to logits and the probabilities $\mathbf{l} \in \mathbb{R}^{G \times V}$ for $v \in V$ entry of codebook $g \in G$. is chosen for each entry Encoder output $\mathcal{Y}$ is also the input for the Transformer to create **contextualised representations** $g : \mathcal{Y} \mapsto \mathcal{Z}$.

The pretraining of the model is inspired by the MLM in BERT explained in 2.3.1, where a certain proportion of time steps are masked from encoder output $\mathcal{Y}$. The masked feature vectors are replaced with one shared trained feature vector. Inputs $\mathcal{Q}$ of quantised $\mathcal{Y}$ are not masked. The final model is then finetuned on the labelled data. The objective function of Wav2vec2 is sum of *Contrastive Loss* and *Diversity Loss*: $\mathcal{L} = \mathcal{L}_c + \alpha\mathcal{L}_d$, where α is a tunable hyperparameter [4].

Contrastive Loss $\mathcal{L}_c$ is computed similarly as explained for Wav2vec in section 3.2. The goal is to distinguish the correct quantised latent audio representation $\mathbf{q}_t \in \mathcal{Q}$ at masked time t of $N + 1$ of quantised candidates $\tilde{\mathbf{q}} \in \mathcal{Q}_t$, $\mathbf{q}_t \in \mathcal{Q}_t$ from a set of distractors κ for each masked time step, $|\kappa| = N$:

$$\mathcal{L}_c = -\log \frac{\exp\left(\operatorname{sim}\left(\mathbf{z}_t, \mathbf{q}_t\right)/\kappa\right)}{\sum_{\tilde{\mathbf{q}} \sim \mathcal{Q}_t} \exp\left(\operatorname{sim}\left(\mathbf{z}_t, \tilde{\mathbf{q}}\right)/\kappa\right)}, \tag{3.2}$$

where $\mathbf{z}_t$ is the output of the contextual network $\mathcal{Z}$ at the centred time step over the masked time step t and *sim* is cosine similarity $\operatorname{sim}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T \cdot \mathbf{v} \,/\, (\|\mathbf{u}\|\|\mathbf{v}\|)$. Distractors κ are uniformly sampled from the same utterance but from other masked time steps.

Diversity Loss $\mathcal{L}_d$ increases the use of the quantised codebook representations. The purpose is to prompt the model to use all entries in the codebook equally often. To ensure that each of the G codebooks has an equal usage of the V entries, $\mathcal{L}_d$ maximises the entropy H of the Gumbel-Softmax distribution[3] for each codebook $\bar{p}_g$ in a batch of utterances. $\mathcal{L}_d$ is obtained as shown in equation 3.3 [4].

[1] Gaussian Error Linear Unit (GELU) is an activation function $x\Phi(x)$, where $\Phi(x)$ is the standard Gaussian cumulative distribution function. Consequently, the GELU can be thought of as a smoother ReLU [20].

[2] Product quantisation is a method for approximate nearest neighbour search that reduces the memory usage and the computational cost of comparing high-dimensional vectors. It works by decomposing the vector space into a Cartesian product of low-dimensional subspaces and quantising each subspace separately [23].

[3] Gumbel-Softmax is a continuous distribution that can approximate categorical samples, and whose parameter gradients can be easily computed [22]

$$\mathcal{L}_d = \frac{1}{GV} \sum_{g=1}^{G} -H\left(\bar{p}_g\right) = \frac{1}{GV} \sum_{g=1}^{G} \sum_{v=1}^{V} \bar{p}_{g,v} \log \bar{p}_{g,v} \tag{3.3}$$

The Wav2vec2 architecture is shown in Figure 3.2.

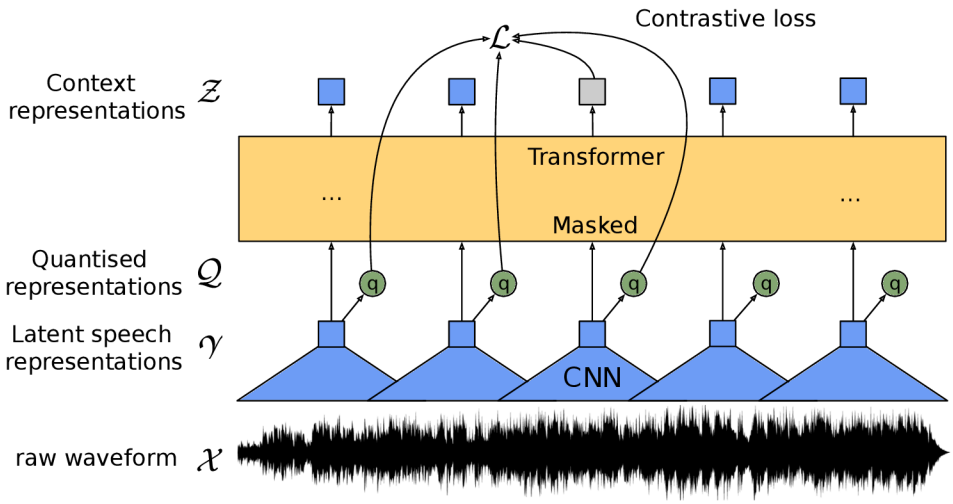


Figure 3.2: Wav2vec2 framework illustration. Wav2vec2 is designed to learn contextualised speech representations and a set of discretised speech units together. Taken from [4].

3.4 HuBERT

Hidden-Unit BERT (HuBERT) [21] is another self-supervised approach for learning speech representation. HuBERT is a BERT-based approach which generates noisy labels (pseudo targets) from an offline clustering for pretraining – it is based on masking continuous speech features to predict cluster assignments. The main idea behind the proposed architecture is that it emphasises the importance of consistency of the targets in addition to the correctness of the targets. This focus helps HuBERT model the sequential structure of the input.

HuBERT follows a similar architecture as Wav2vec as it uses *CNN encoder* followed by *BERT* encoder. However, the pretraining process differs. HuBERT does not use any quantisation module to the output of the CNN encoder. HuBERT instead creates pseudo targets – *hidden units*, by a clustering process from a raw audio.

Hidden units are frame-level targets obtained from the waveform. Let $\mathbf{X}$ be the input raw audio with speech of S frames, $[\mathbf{x}_1, \ldots, \mathbf{x}_S] = \mathbf{X}$. Let Y be MFCCs features extracted from frames of X. Then the pseudo targets are estimated as $h(\mathbf{Y}) = [z_1, \ldots, z_T]$, where $h(\mathbf{Y})$ is a clustering algorithm, for instance *k-means* and $z_i \in \{1, \ldots, C\}$ are class categorical variable of C classes.

The next step follows the original BERT by using the MLM objective for training. Around 50 % of transformer encoder input features from CNN encoder are masked, where p % of the time steps are selected randomly as start indices and spans of l steps are masked. Instead of contrastive and diversity loss as Wav2vec, HuBERT uses *cross-entropy* loss over masked $\mathcal{L}_m$ and unmasked $\mathcal{L}_u$ time steps. $\mathcal{L}_m$ is defined as:

$$\mathcal{L}_m = \sum_{t \in M} \log p_f \left(z_t \mid \tilde{W}, t \right) , \tag{3.4}$$

where M is the set of masked time steps, $\tilde{W}$ is the edited sequence of length T, where $w_t , t \in M$ is replaced with a mask embedding $\tilde{w}$. a distribution over the targets at each time step is defined as $p_f(\mathbf{z}_t|\tilde{W}, t)$. $\mathcal{L}_u$ is obtained accordingly, but we sum over time steps which are unmasked. Then, the final loss is obtained as:

$$\mathcal{L} = \alpha \mathcal{L}_m + (1 - \alpha)\mathcal{L}_u , \tag{3.5}$$

where α is a tunable parameter from 0 to 1. HuBERT authors claim $\alpha = 1$ forces the model to learn the acoustic representation of unmasked segments and also the speech data's long-term temporal structure. In addition, this setup brings better quality to the targets from clusters [21].

HuBERT proposed an idea for improving the quality of cluster targets by using various clustering methods – e.g. creating k-means models ensemble. Let $Z^{(k)}$ be the target sequences generated by the k-th clustering model. Then loss $\mathcal{L}_m$ can be written as:

$$\mathcal{L}_m = \sum_{t \in M} \sum_{k} \log p_f^{(k)} \left(z_t^{(k)} \mid \tilde{W}, t \right) . \tag{3.6}$$

$\mathcal{L}_u$ is obtained accordingly by taking unmasked time steps.

HuBERT iteratively refines cluster assignments. a new generation of clusters can be obtained by training a discrete latent model on the learned representations. Then the learning process continues with the newly found units.

Encoder proceeds to parametrise the distribution $p_f^{(k)}(c|\tilde{W}, t)$ by:

$$p_f^{(k)}(c \mid \tilde{W}, t) = \frac{\exp\left(\text{sim}\left(A^{(k)} o_t, e_c\right)/\tau\right)}{\sum_{c'=1}^{C} \exp\left(\text{sim}\left(A^{(k)} o_t, e_{c'}\right)/\tau\right)} , \tag{3.7}$$

where A is the projection matrix, e_c is the embedding of class c, sim is the cosine similarity between two vectors, and τ scales the logit. When cluster ensembles are used, one projection matrix $A^{(k)}$ is applied for each clustering model k. The architecture is depicted in in Figure 3.3

HuBERT shows better performance for finetuning ASR systems than Wav2vec. In addition, audio-to-vector representations generated by HuBERT are robust enough even for other downstream tasks, e.g. generative tasks [21].

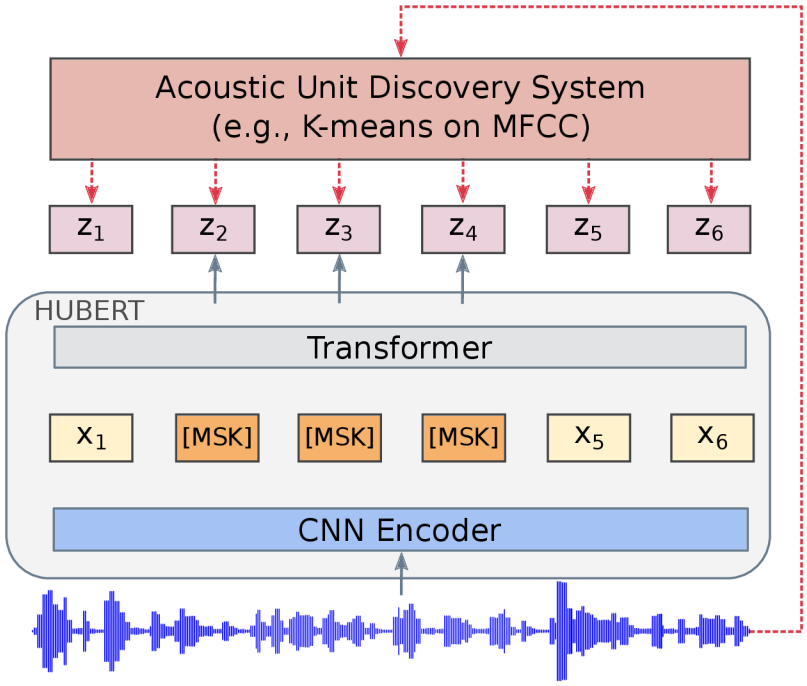


Figure 3.3: HuBERT model illustration. HuBERT is designed to predict hidden cluster assignments of the masked frames, generated by k-means clustering. Taken and vectorised from [21].

Chapter 4

Joint-representation models

Self-supervised learning has the same general idea for different modalities, but the specific algorithms and objectives vary a lot as they are designed for a given modality. Joint-representation models create such semantic subspace, that is shared for all the modalities. This is done to capture the relationships and interactions between these different types of input. This thesis focuses on models that use both speech and text modalities to create a joint representation. Several architectures relevant to this thesis are for example *SpeechT5* model developed by Ao J., Wang R. [2], *Data2vec* by Baevski A. which is also using images as additional modality [3] or *SeamlessM4T* by Meta [9].

4.1 SpeechT5

SpeechT5 is a multimodal speech/text encoder-decoder model. SpeechT5 evolved over ideas from text and speech state-of-the-art like BERT [24], Wav2vec [4] or HuBERT [21]. However, the main idea stems from the Text-To-Text Transfer Transformer (T5) method [39]. SpeechT5 aims to range of text/speech downstream tasks than just ASR, but also speech translation, speech identification, text-to-speech, voice conversion, and speech enhancement [2].

The goal of SpeechT5 is to use an encoder-decoder framework for every spoken language processing task, whether it involves speech or text as input or output. Then the same pretrained model can be applied with bimodal data to different tasks.

To achieve this, the input for downstream tasks needs to be in the same vector space. This text and speech mapping into shared quantisation space is accomplished by pre-processing by several pre-nets according to the modality. To learn better cross-modal features the quantised latent representations are randomly mixed together with the contextual states. Then it is connected to the encoder-decoder backbone which ensures sequence-to-sequence conversion and then post-nets are applied accordingly to the required modality [2].

4.1.1 SpeechT5 encoder

For this thesis, the crucial part is the encoder and the multimodal mapping into joint vector space. The architecture of the SpeechT5 encoder is built by the `i)` speech pre-net, `ii)` text pre-net – producing discrete tokens which are shared for the `iii)` quantiser – capturing the modality-invariant information. The architecture is depicted in Figure 4.1.

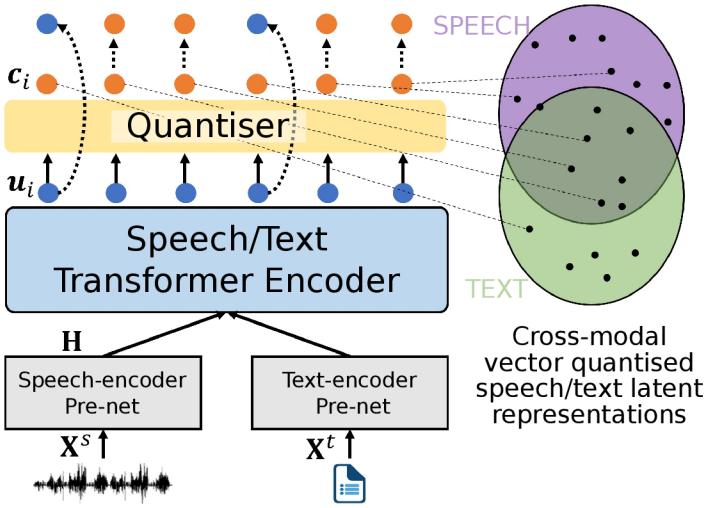


Figure 4.1: SpeechT5 multimodal encoder architecture. Vectorised from [2].

i) Speech pre-net The component has *Wav2Vec2* [4] convolutional feature extractor as pre-net (explained in section 3.3). Let's assume data points $\mathbf{X}_{sp}$ of dataset $\mathcal{D}$. The raw recordings of $\mathbf{X}_{sp}$ are downsampled by Wav2vec2 to produce sequence encoded of speech utterances $\mathbf{H} = (\mathbf{h}_1, \ldots, \mathbf{h}_N)$. The Speech pretraining utilises unlabelled data from $\mathcal{D}$ to learn general speech representations with self-supervised techniques. Training is based on *HuBERT* [21] (explained in section 3.4), where acoustic hidden units provide the frame-level targets $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_N)$. Let $\hat{\mathbf{H}}$ be the masked pre-net output $\mathbf{H}$, where 8 % of time-steps are selected randomly as start indices and span mask approaches of size 10 timesteps are applied. From the masked input $\hat{\mathbf{H}}$, HuBERT based encoder produces hidden representations $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_N)$. Over the masked timesteps in $\hat{\mathbf{H}}$ the cross-entropy loss $\mathcal{L}_{ce}$ is calculated as follows:

$$\mathcal{L}_{ce} = \sum_{n \in \mathcal{M}} \log p\left(\mathbf{z}_n \mid \hat{\mathbf{H}}, n\right) , \tag{4.1}$$

where $\mathcal{M}$ is the set of the masked timesteps from $\hat{\mathbf{H}}$ and $\mathbf{z_n}, \mathbf{z_n} \in \mathbf{Z}$ is the frame-level target at the corresponding timestep n [2].

The final loss for the whole speech part module can be computed as follows:

$$\mathcal{L}_{sp} = \mathcal{L}_{ce} + \mathcal{L}_1 + \mathcal{L}_{bce} , \tag{4.2}$$

where $\mathcal{L}_1$ is the L_1 distance between the original data $\mathbf{X}_{sp}$ and the reconstruction of the original data which were obtained by using the randomly masked input. $\mathcal{L}_{bce}$ is binary cross-entropy loss for the `stop` token.

ii) Text pre-net SpeechT5 uses shared embeddings for both text-encoder pre-net and text-decoder pre/post-nets. The pre-net converts a token index into an embedding vector, while the post-net turns the hidden state into a probability distribution of tokens, normalised by the softmax function.

Similar to the speech nets training, text pretraining also employs unlabelled data $\mathbf{X}_{txt}$ of dataset $\mathcal{D}$. The training is accomplished by reconstructing the output of the model

$\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_N)$ to the original $\mathbf{X}_{txt}$ data with the use of the corrupted data $\hat{\mathbf{X}}_{txt}$. The corrupted data $\hat{\mathbf{X}}_{txt}$ is produced by using the text spans masking approaches presented in *BART* [27], where 30 % of text spans are randomly chosen to mask of various length from a Poisson distribution of $\lambda = 3.5$. Every span is then substituted with a single masked token. The objective function is maximum likelihood estimation $\mathcal{L}_{mle}$ computed as follows:

$$\mathcal{L}_{mle} = \sum_{n=1}^{N} \log p\left(\mathbf{y}_n \mid \mathbf{y}_{<n}, \hat{\mathbf{X}_{txt}}\right) , \tag{4.3}$$

where $\mathbf{y}_{<n}$ is the text seen before the $\mathbf{y}_n$ [2].

`iii)` Joint vector space mapping Approaches described in `i)` and `ii)` are limited to either speech or text data to model acoustic or linguistic information separately.

To establish a cross-modality mapping between speech and text a cross-modal vector quantisation approach is proposed to create a shared representation that captures modality-invariant information. SpeechT5 authors used a shared codebook to create quantised embeddings for alignment as shown in Figure 4.1. Let $\mathbf{u}_i$ be the continuous representations of speech and text obtained from `i)` and `ii)` modules. The quantiser converts $\mathbf{u}_i$ into discrete representations $\mathbf{c}_i$ by using a fixes-size codebook $\mathbf{C}^K$, where K is the number of learnable embeddings. Therefore, $\mathbf{c}_i$ is obtained by using the nearest neighbour search via the Euclidean L_2 distance between the $\mathbf{u}_i$ and the embedding of each latent code $\mathbf{c}_j \in \mathbf{C}^K$ calculated as

$$\mathbf{c}_i = \min_{j \in [K]} \|\mathbf{u}_i - \mathbf{c}_j\| .$$

Afterwards, 10 % of the contextual representations are replaced with quantised latent representations at corresponding time steps and the cross-attention is computed on the combined representations. This process explicitly directs the quantiser to extract cross-modal information [2].

The final pretraining loss $\mathcal{L}_{fin}$ with unlabeled data of both speech and text is represented as follows:

$$\mathcal{L}_{fin} = \mathcal{L}_{sp}^{(speech)} + \mathcal{L}_{mle}^{(text)} + \gamma \mathcal{L}_d , \tag{4.4}$$

where $\gamma = 0.1$ is set for pretraining. $\mathcal{L}_d$ is the diversity loss obtained by maximising the entropy of the averaged Softmax distribution computed as follows:

$$\mathcal{L}_d = \frac{1}{K} \sum_{k=1}^{K} p_k \log p_k$$

where p_k represents the average probability of selecting the k-th code from the codebook. It is computed as follows:

$$p_k = \frac{1}{N} \sum_{n=1}^{N} softmax(x_{nk}),$$

where N is the number of data points in the batch and x_{nk} is the output for the n-th data point and the k-th code.

4.2 SeamlessM4T

SeamlessM4T (Seamless **M**assively **M**ultilingual & **M**ultimodal **M**achine **T**ranslation) is a multimodal multilingual model developed by Meta in 2023 [9]. SeamlessM4T provides speech-to-speech translation, speech-to-text translation, text-to-speech translation, text-to-text translation, and automatic speech recognition. It supports up to 100 languages.

4.2.1 SeamlessM4T encoder

For training, one million hours of open speech audio data were utilised to learn self-supervised speech representations using w2v-BERT 2.0[1]. a multimodal corpus of automatically aligned speech translations was created – SeamlessAlign[2]. Then the human-labelled data were filtered and merged with the pseudo-labelled data in a total of 406,000 hours.

SeamlessM4T utilises shared multilingual and multimodal embeddings space named SONAR (**S**entence-level multim**O**dal and la**N**guage-**A**gnostic **R**epresentations) proposed by Duquenne et al. 2023 [13]. SONAR presents different methods for encoding text and speech. The architecture is shown in Figure 4.2.

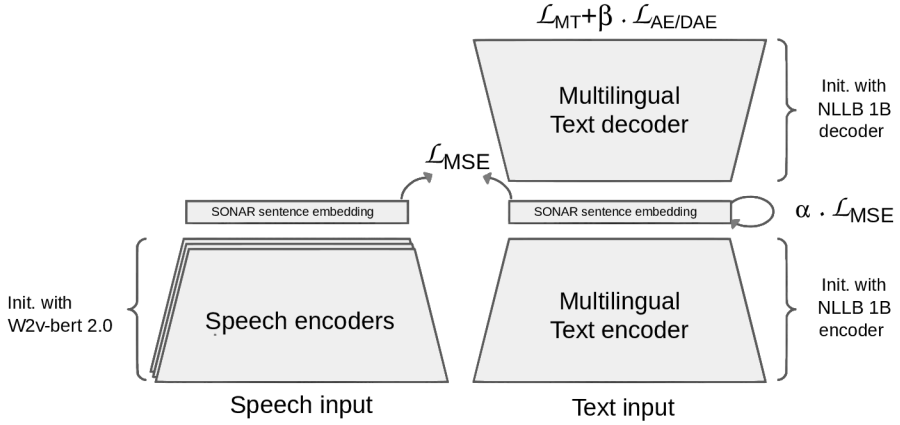


Figure 4.2: SeamlessM4T encoder architecture for shared embedding space of both text and speech – SONAR [9]. It is trained with a combination of machine translation loss, mean square error loss and auto-encoder loss as denoted in equation 4.5. Taken and vectorised from [13].

Text modality Data preparation for text follows *NLLB* dataset[3] and utilises the topline NLLB 1B model for initialisation.

In text encoding, an encoder-decoder approach is utilised to learn sentence embeddings with a translation objective, computed thanks to the additional decoder. This approach differs from the sequence-to-sequence model in the bottleneck layer and the pooling function

[1]W2v-BERT 2.0 is based on w2v-BERT [7] which combines contrastive learning of Wav2vec and masked prediction learning of BERT. W2v-BERT 2.0 enhances the original w2v-BERT by additional codebooks into both of its learning objectives.

[2]SeamlessAlign is the large open dataset for multimodal translation – totalling 470,000 hours. The dataset has automatically created alignments for speech and text multilingual data [9]. It is developed by Meta with a freely available reconstructible recipe at https://tinyurl.com/27p6vphp .

[3]NLLB (**N**o **L**anguage **L**eft **B**ehind) dataset for machine language translation with more than 200 languages and it also includes low-resource languages [32]. Accessible at https://tinyurl.com/25byfb5o .

which computes a fixed-size sentence representation between the encoder and the decoder by pooling the token-level outputs of the encoder. Rather than performing cross-attention over a variable-length sequence of encoder outputs, the decoder focuses on this single vector during each step of decoding. The translation objective decoder employs a finetuning method called random interpolation decoding. It is based on an encoder-decoder model with a bottleneck representation. In this approach, the encoder weights are frozen, and only the decoder weights are finetuned for a specific task: given a bilingual text pair $\mathbf{x}$, $\mathbf{y}$, both $\mathbf{x}$ and $\mathbf{y}$ are encoded using the frozen encoder. Then, a new embedding $\mathbf{z}$ is generated as a random interpolation of the embeddings of $\mathbf{x}$ and $\mathbf{y}$, and the model is trained to decode this interpolated embedding $\mathbf{z}$ back into $\mathbf{y}$. This method effectively blends translation with auto-encoding[4].

Audio modality Data preparation for audio gathers publicly available repositories of crawled web data first. Pre-processing involves resampling of crawled data to 16 KHz. Afterwards, the non-speech data with the Audio Event Detection (AED) model were filtered out. Recordings were also split into smaller chunks mapped closely to contain a pseudo sentence, similar to sentences in text corpora. The length of each segment corresponds to that of a typical sentence. This automatic sentence-segmentation algorithm is proposed in Duquenne's work [12].

Multilingual sentence representations for speech are trained with a teacher-student method. The teacher model is an encoder for multi-lingual sentence embeddings trained on text which converts embeddings into new trained text sentence embedding space.

The student is using a speech encoder with w2v-BERT 2.0, optimised for 143 languages, which encodes audio and the output is transformed into fixed-size representations by using Attention-pooling. It is trained to minimise the Mean Squared Error (MSE) loss between the transcription sentence embeddings and the speech sentence embeddings. Thanks to MSE loss the SONAR text encoder could be used on input for speech. Data used for training were also manually transcribed datasets for ASR, with collected at least 100 hours per language. Languages of the same linguistic family were grouped and the groups were trained together in one speech encoder. As the Sonar embedding space comes with a text decoder, the speech encoder is evaluated by the individual speech encoders on a speech-to-text-translation task.

The SONAR training combines the translation loss $\mathcal{L}_{MT}$, the auxiliary MSE loss $\mathcal{L}_{MSE}$ and the denoising auto-encoding loss $\mathcal{L}_{AE/DAE}$, to create the SONAR embedding space. It is computed as:

$$\mathcal{L} = \mathcal{L}_{\text{MT}} + \alpha \cdot \mathcal{L}_{\text{MSE}} + \beta \cdot \mathcal{L}_{\text{AE/DAE}}\,, \tag{4.5}$$

where α is set to 0.1 and β to 0.01. The whole SeamlessM4T architecture with embedded SONAR encoder is shown in Figure 4.3.

[4]Auto-encoder is an approach used to learn efficient representation of unlabelled data. An auto-encoder learns two functions: an encoding function that transforms the input data, and a decoding function that reconstructs the input data from the encoded representation. Auto-encoders have the advantage of encouraging the encoding of fine-grained details of the input. However, this objective alone is unlikely to learn the proper semantic representation of sentences. It is simpler to learn than a translation objective [9].

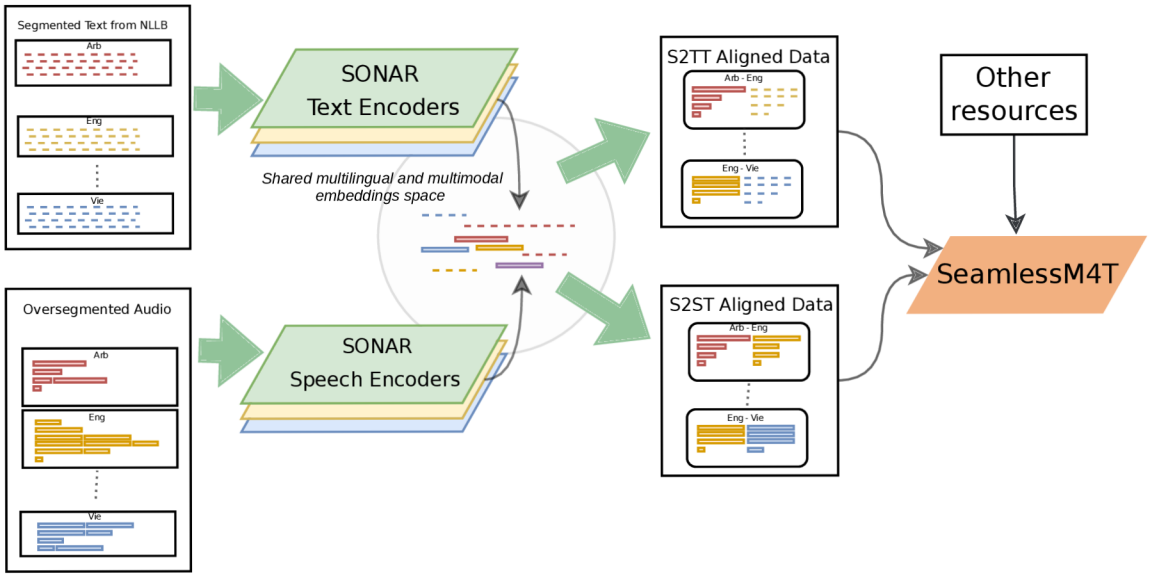


Figure 4.3: SeamlessM4T training with SONAR encoder to train for speech-to-text-translation and speech-to-speech-translation pairs. Taken from [9].

Chapter 5

Datasets

This chapter will describe the datasets required for the evaluation of the proposed methods. Since the thesis is in the text and speech domain, naturally datasets for automatic speech recognition were chosen. One of the most used datasets in the evaluation of ASR systems is *LibriSpeech* [35]. However, the dataset that more reliably simulates the real use case of semantic search is the Fisher dataset.

5.1 LibriSpeech

The *LibriSpeech* corpus is derived from audiobooks that are part of the *LibriVox*[1] project, and contains 1000 hours of speech sampled at 16 kHz. LibriSpeech is freely available[2] under CC BY 4.0 license. The dataset contains in total 1000 hours of speech sampled at 16 kHz. The data are only in English. The dataset is equally balanced among genders and has uniform per-speaker durations. The recordings are short ≈ 30 seconds long utterances. The split of LibriSpeech to subsets is presented in Table 5.1. For this thesis, subsets `train-clean-360` and `dev-clean` are chosen. Most of the experiments were processed with *Librispeech's* `dev-clean` subset for its size and flexibility and `train-clean-360` served mainly during data preparation for finetuning.

Subset	Hours	Minutes per speaker	Speakers (female)	Speakers (male)	Speakers (total)
dev-clean	5.4	8	20	20	40
test-clean	5.4	8	20	20	40
dev-other	5.3	10	16	17	33
test-other	5.1	10	17	16	33
train-clean-100	100.6	25	125	126	251
train-clean-360	363.6	25	439	482	921
train-other-500	496.7	30	564	602	1166

Table 5.1: Subsets of LibriSpeech dataset [35].

Librispeech is delivered with metadata about the speakers like name, sex, total spoken speech length, and each is given a specific ID. The given transcripts are raw. They are given per file without any timestamps. Also, no additional tags like `[laugh]`, `[sigh]` and

[1] https://librivox.org/ – LibriVox is a global community of volunteers who record audiobooks from public domain texts and make them available for free on the internet.

[2] https://www.openslr.org/12 – LibriSpeech ASR corpus with all available subsets for download.

similar tags are used – which is natural given how the data was obtained. Given that the thesis aims to evaluate innovative methods for a keyword spotting system and to search for semantically similar expressions, the raw transcripts are not sufficient. It is required to have at least timestamps per word. Therefore forced alignment techniques are applied to the Librispeech dataset as explained in section 5.3.

5.2 Fisher

Fisher English Training Speech dataset was developed by the Linguistic Data Consortium (LDC)[3]. The Fisher dataset closely resembles real telephone conversations, making it suitable for testing and building a working system. It comes in two parts and contains 1,960 hours of English conversational telephone speech (CTS), in a total number of 11,699 recordings. The gender distribution for the entire collection of participants makes in total of 6,813 females and 5,104 males.

The Fisher telephone conversation collection was created to build robust automatic speech recognition (ASR) systems. Fisher data creators asked a large number of participants to make up to 10-minute calls with varied topics.

Fisher is distributed with additional information regarding the speakers involved and the types of telephones used. It is provided with detailed precise transcriptions including tags for emotions, sighs, etc. The audio files are presented in NIST SPHERE format and contain two-channel mu-law data sampled at 8000 Hz [8].

Fisher subset As the Fisher dataset contains too many recordings, a subset was created for efficiency in development and evaluation. This subset consists of 60 recordings of telephone conversations, 10.12 hours in total. The 60 recordings were chosen accordingly:

- 30 recordings are from the Fisher part 1 and the other 30 from Fisher part 2
- The 30 recordings within the part of the dataset consisted of 10 recordings from purely male calls, 10 from purely female calls, and the remaining 10 mixed.
- The recordings were chosen uniformly.

Fisher metadata comes with aligned phrases to audio, however, it uses a different format than STM and it does not provide alignments per word. Therefore the same technique as for Librispeech – forced alignment is applied as explained in the following section 5.3.

Additional changes to the transcripts were applied according to the used model, as each tokeniser expects different input tokens. This mainly consists of deleting tags in transcripts about non-verbal communication, some diacritics, or transcribing numbers into text form. However, models utilised throughout the thesis required generally specific modifications namely:

- Upsampling – all of the models used require 16 kHz recordings thence all data needs to be upsampled from 8 kHz to 16 kHz
- Channels – When processing audio, models require mono channel audio on input. Since Fisher data are telephone conversations, they are stereo tracks having separate

[3]Fisher dataset is available under LDC subscription: https://catalog.ldc.upenn.edu/LDC2004S13

channels for caller and receiver. Two solutions are performed: `i)` splitting stereo channels into two mono channels `A`-left channel and `B`-right channel and `ii)` merging two stereo tracks into one mono channel track. The first approach doubles the recordings of the dataset, while the second method affects the original audio because of the merge.

- Audio format – NIST `.sph` (SPHERE) format is not a widely used audio format, therefore all of the audios are converted to a more typical format for easier processing – FLAC[4].

5.3 Forced Alignment

For such scenarios when we have audio and the corresponding transcript, a method known as *forced alignment* is utilised for creating timestamps of the corresponding text within the audio. Forced alignment could be done with several techniques like Dynamic Time Warping (DTW) [31], Hidden Markov Model (HMM) [45] or Deep Neural Networks (DNN), Recurrent Neural Net (RNN) used for example in the Kaldi toolkit[5]. However, the technique used for forced alignment is based on Wav2Vec2 from section 3.3. The method is based on CTC-segmentation [25]. The CTC segmentation process involves three main steps:

1. Forward propagation: Probabilities for each character at every time step are obtained from Wav2vec2 output. These probabilities are mapped to a trellis diagram.
2. Backtracking: Starting with the time step that has the highest probability for the last character, backtracking identifies the most likely sequence of characters across all time steps.
3. Confidence score: This method produces a probability for each aligned character or word, allowing for the derivation of a confidence score for each utterance. This score helps in identifying and filtering out utterances that are likely misaligned.

Since Wav2Vec2 improves ASR tasks it is also performing sufficiently for forced alignment. The used format for forced aligned transcriptions is‘ in *STM* format. The *STM* format contains columns with base filename, channel, speaker ID, the start of the segment in seconds and end segment timestamp in seconds, the full path of the input audio and finally the keyword – the transcript within the segment. An example of an STM file on the LibriSpeech dev-clean subset is as follows:

```
1272-128104-0002 A 128104 0.60 0.70 </dev-clean/audio/1272-128104-0002.flac>  HE
1272-128104-0002 A 128104 0.76 1.00 </dev-clean/audio/1272-128104-0002.flac>  TELLS
1272-128104-0002 A 128104 1.12 1.22 </dev-clean/audio/1272-128104-0002.flac>  US
1272-128104-0002 A 128104 1.40 1.58 </dev-clean/audio/1272-128104-0002.flac>  THAT
1272-128104-0002 A 128104 1.72 1.80 </dev-clean/audio/1272-128104-0002.flac>  AT
1272-128104-0002 A 128104 1.84 2.02 </dev-clean/audio/1272-128104-0002.flac>  THIS
1272-128104-0002 A 128104 2.14 2.56 </dev-clean/audio/1272-128104-0002.flac>  FESTIVE
1272-128104-0002 A 128104 2.62 3.08 </dev-clean/audio/1272-128104-0002.flac>  SEASON
1272-128104-0002 A 128104 3.14 3.22 </dev-clean/audio/1272-128104-0002.flac>  OF
1272-128104-0002 A 128104 3.26 3.36 </dev-clean/audio/1272-128104-0002.flac>  THE
1272-128104-0002 A 128104 3.40 3.66 </dev-clean/audio/1272-128104-0002.flac>  YEAR
```

[4]FLAC – **F**ree **L**ossless **A**udio **C**odec, is an audio coding format for lossless compression developed by the non-profit organisation – Xiph.Org Foundation.

[5]Kaldi Speech Recognition Toolkit: https://kaldi-asr.org/

Additional metadata with timestamps per word is necessary to properly evaluate the KWS performance.

Chapter 6

Evaluation

This chapter presents evaluation metrics and an approach for measuring the quality of the developed systems. This part is crucial for comparing the performance of the novel approaches to the legacy technologies, for example, keyword spotting (KWS) also known as spoken term detection.

In the sections Keyword spotting evaluation 6.1 and Semantic search evaluation 6.2, we present the tasks used for evaluation. There are 4 tasks in total:

1. **Exact KWS**: This task is the standard keyword spotting task. It searches for entered input queries literally.
2. **Fuzzy KWS**: This task extends the Exact KWS task by allowing some Levenshtein distance between the matching strings.
3. **KWS + synonyms**: The task involves thesaurus which extends Exact KWS by including all grammar forms of the keywords and their synonyms.
4. **Semantic phrases search**: The task objective is to match semantically similar phrases.

Tasks 1, 2 and 3 share the same list of input queries for evaluation. The list includes queries as follows:

```
start
meeting
suspect
place
activity
smoking
drug
```

However, the evaluation sets are changed which are expanded according to the task. Task 4 uses both input query phrases and the evaluation set independently.

The ground truths for tasks 1 and 2 can be estimated automatically, while tasks 3 and 4 require manual intervention.

6.1 Keyword spotting evaluation

Evaluating the KWS task is more straightforward compared to the semantic search task. The objectivity stems as it searches for specific keywords or phrases, eliminating any ambiguity. This makes it easy to distinguish between correct and incorrect results.

Since KWS is not a novel technology, it can be compared to existing systems. The chosen system for comparisons is the 6th generation Keyword Spotting by the Phonexia company[1].

Ground truths TP, TN, FP and FN are calculated using the inference outputs of the models for the given queries and using the STM file, which provides per-word alignments to audio. To evaluate KWS tasks 1 and 2, two strategies were implemented for estimating the correct search results for the given shared list of input queries list.

The first is the strict one (referred to as *Exact KWS*), to address task 1. The ground truth search results are the literal exact matches of the query within the given STM file. This approach reflects standard KWS.

The second is more benevolent (referred to as *Fuzzy KWS*), to address task 2. The ground truth search results are obtained by comparing the query to the keywords within the STM file with the fuzzy string matching[2]. This method allows for a slightly broader variability. For instance, for the search query "start", cases like "starts", "started", or "starting" will be accepted. However, cases like "smart", and "tarts" will also be matched.

The output produced from the proposed systems is scored for consecutive time frames. The size of the time frame varies depending on the query size, the model used, and modifications to the search algorithm. Therefore, in order to accept results where a correctly found case does not align exactly with the ground truth time segment, a relaxed collar is used. This approach is illustrated in Figure 6.1.

6.1.1 Evaluation metrics

One of the commonly used evaluation metrics is standard *Equal Error Rate* (EER) [15] and measures like *precision* and *recall*. Commonly used for comparing the quality of the proposed systems are *Detection Error Trade-off* (DET) curves, together with *Receiver Operating Characteristic* (ROC) curves.

EER *Equal error rate* represents the percentage for the given threshold θ_{EER} where the number of false negatives – (FN)[3] are equal to the same amount of false positives (FP)[4]. The lower the equal error rate value is, the higher the accuracy of the system [15].

While Equal Error Rate (EER) is a commonly used metric in general, it may not be ideal for Keyword Spotting (KWS) due to varying recording lengths. This is because the number of false acceptances is proportional to the length. Therefore, the length normalisation is required. However, in our specific case, the datasets used have similar lengths, which should

[1]The Phonexia Keyword Spotting system can utilise both query-by-example and text-based queries. More details at https://tinyurl.com/23g64hdd

[2]Fuzzy string matching is a method of approximate string matching by using Levenshtein distance [50].

[3]False negative is also known as *missed detection*, *underestimation* or *false rejection*. It is the error of the incorrectly rejected sample.

[4]False positive is also known as *false alarm*, *overestimation* or *false acceptance* – the error of incorrectly accepted samples.

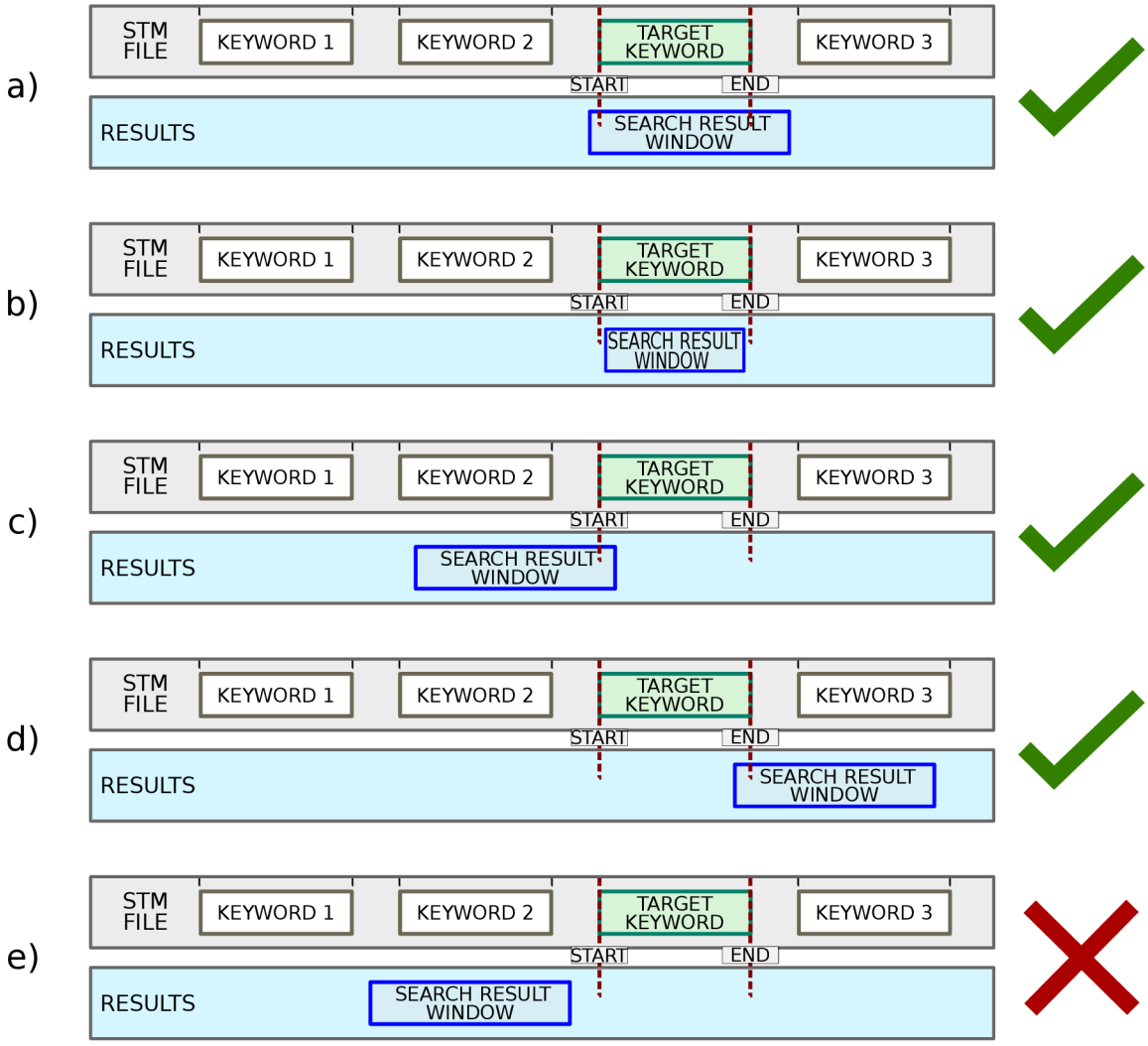


Figure 6.1: The Figure presents the relaxed collar. The Figure shows scenarios when the automatic evaluation accepts the search result as correct results – scenarios `a)`, `b)`, `c)`, `d)`. The scenario when the result is not accepted is depicted in `e)`. Thanks to the proposed relaxed collar, it is enough to interfere anyhow with the target keyword to accept the result.

not lead to an imbalance. Particularly, we are comparing the created systems to each other relatively. Therefore, it is important to follow the relative improvement of EER.

As the EER is a pooled metric, just one global threshold θ_{EER} is applied for all queries. The metric is defined as:

$$\mathrm{EER} = \frac{\sum_{Q\in\Delta} N_{target}(Q) - N_{TP}(Q,\theta_{EER})}{\sum_{Q\in\Delta} N_{target}(Q)}\,, \tag{6.1}$$

where Q is the query, Δ is the set of all queries and N_{target} is the number of the target search results. The following conditions have to be satisfied:

$$\sum_{Q\in\Delta} N_{target}(Q) - N_{TP}(Q,\theta_{EER}) = \sum_{Q\in\Delta} N_{FP}(Q,\theta_{EER})\,, \tag{6.2}$$

where N_{TP} is the number of true positives (TP)[5] and N_{FP} is the number of false positives [15].

EER can also be interpreted as the point at which the False Acceptance Rate (FAR) curve and False Rejection Rate (FRR) curve intersect. FAR and FRR are computed as follows:

$$\begin{aligned} \text{FAR} &= \frac{FP}{FP + TN} \\ \text{FRR} &= \frac{FN}{FN + TP}\,, \end{aligned} \tag{6.3}$$

where TN is the count of true negatives.

Precision and recall The Phonexia's KWS system is also evaluated with *precision* and *recall*. Precision explains what is the proportion of those identified truly correctly (TP) to positive identifications ($TP + FP$). Recall shows the proportion of the cases identified correctly (TP) to the positive cases ($TP + FN$). Precision and recall are calculated as follows:

$$\begin{aligned} \text{Precision} &= \frac{TP}{TP + FP} \\ \text{Recall} &= \frac{TP}{TP + FN}\,. \end{aligned} \tag{6.4}$$

Precision and recall are computed at a specific operation point. The setting depends on the actual use case and the cost of the FP error, respectively the cost of the FN error. In the context of KWS, higher precision is more beneficial when the cost of the FP is high – the better the precision, the fewer irrelevant search results occur - less false alarms. The higher recall is more suitable when the FN cost is high, which means that the higher the recall, the fewer relevant search results are missed. Phonexia KWS system is calibrated towards the higher recall, as the cost of the miss is high in the domain.

DET *The Detection Error Trade-off* (DET) is a widely used method to display the performance of systems across different operational points. It plots the FN probability (P_{FN}) and FP probability (P_{FP}) on its axes, providing insights into both types of errors. Optimal system performance is indicated by a DET curve that approaches the lower left corner [28].

For a given threshold θ DET is defined as a dependency of FN probability $p_{FN}(\theta)$ and FP probability $p_{FP}(\theta)$. DET curve does not give a single value but provides a comprehensive graph depicting the system's performance across various operating points [15].

ROC A *receiver operating characteristic* (ROC) curve is similar to DET curves. However, the ROC curve represents the plot of the TP probability (P_{TP}) against the FP probability (P_{FP}) at various operating points.

Optimal system performance is depicted by the ROC curve approaching the top left corner. The *Area Under the ROC Curve* (ROC AUC), which ranges from 0 to 1 and is a real number, quantifies the system's performance. Therefore, the ideal cut-off value is the

[5] True positive is also known as *hit* or *true acceptance* – a correctly found sample.

one that achieves the highest true positive rate (to 100 %) [14] and the lowest false positive rate (to 0 %).

MinDCF *Minimum Detection Cost Function* (MinDCF) depicts the operation point where the cost of the error is minimal. MinDCF shows additional information to other metrics as it involves the imbalances of the number of target samples and non-target samples [33]. It is defined as a weighted sum of the FN probabilities $\alpha(\theta)$ and FP $\beta(\theta)$ at a given threshold θ:

$$DCF(\theta) = C_{FN} \times \alpha(\theta) \times P_{tar} + C_{FN} \times \beta(\theta) \times P_{nontar}\,. \tag{6.5}$$

The parameters C_{FN} and C_{FN} are the relative costs of detection errors, and the parameter P_{tar} is the a-priori probability of the target and $P_{nontar} = 1 - P_{tar}$ [47]. It is usual to indicate $MinDCF$ on DET curves to show the MinDCF operation point.

6.2 Semantic search evaluation

The KWS evaluation approach measures only a fraction of the capabilities of the proposed system. Unfortunately, the evaluation for semantically similar matches is not as straightforward as for the KWS evaluation. The capabilities are captured by tasks 3 and 4 – "*KWS+ synonyms*" and "*Semantic phrases search*" tasks.

For example, let us assume the following keyword "start". Semantically similar words to "start" could be keywords like "beginning", "kickoff", "opening", "initiate" or "launch". To get this similar meaning it is sufficient to look up the thesaurus.

However, things become more ambiguous for phrases where looking up synonyms may not be sufficient. For example, consider the following phrases: "I had an exam at the university" and "I took a test at the faculty". Are these phrases semantically close enough to be considered valid search results? Or consider another example: "I had an exam at the university", "They had written a test at the elementary school", and "The teacher corrected the tests at the high school." Are these examples close enough to be accepted? These are the questions that are crucial for creating a functional system. The main problem is that it is difficult to objectively evaluate the correct answers. While some users may consider the second example as a close result, some may not even accept the first example.

To solve this ambiguity evaluation sets with manually labelled ground truths for the predefined list of queries are created. The first evaluation set is based solely on keywords with synonyms from the thesaurus. The second evaluation set consists of manually labelled phrases which are considered to be semantically close enough to the predefined query phrase.

With these evaluation sets, the results are then measured analogously to the standard KWS tasks (1 and 2). In addition, the same relaxed collar as presented in Figure 6.1 is used.

6.2.1 Evaluation set with synonym keywords

This manual ground truth aims to increase the relevance of the objective evaluation for the semantic search tasks. The ground truth for this evaluation set is labelled within *Librispeech dev-clean* subset. The created set evaluates the *KWS+ synonyms* task.

Initially, we took the shared input queries list, with keywords as follows: "start", "meeting", "suspect", "place", "activity", "smoking", "drug". Then, thanks to the created STM

file as proposed in section 5.3, we can choose the segments with the chosen keywords including all of the grammatical forms of the keyword. Afterwards, for each keyword, we find synonyms, by using the English Thesaurus by Cambridge University[6], and search for all grammatical forms in STM file segments.

For example, the keyword "smoking" is found only once in the Librispeech dev-clean subset. However, synonyms like "burning" appears 6 times, "blazing" and "fiery" 2 times, "kindled" and "glowing" once.

Therefore, we expect that the false acceptance errors detected in standard KWS evaluation with high match scores should belong to semantically similar terms, such as synonyms.

6.2.2 Evaluation set with semantically similar phrases

Analogous to keywords, an evaluation set with ground truths with semantically similar phrases is proposed. This evaluation set is used to measure task 4 – *Semantic phrases search*.

Initially, we selected three sentences from the Librispeech dev subset as predefined input query phrases: `i)` "It was the worst Sunday", `ii)` "The wind was so strong", `iii)` "I had much pleasure in reading". These sentences were chosen for their general meaning, which they express: `i)` unpleasant experience, `ii)` weather related, outdoor conditions, `iii)` joy from something. Then these sentences are also the ground truth target phrases which need to be found.

Next, semantically similar phrases to the predefined input queries were chosen. Unfortunately, selecting the phrases within the STM file of Librispeech cannot be accomplished using a thesaurus directly. While dictionaries may provide synonyms, the Librispeech subset is sufficiently small to allow for matching similar sentences with only one or two-word changes. While dictionaries may provide synonyms, simply replacing words with their synonyms is not enough to find similar sentences within a text. In the end, the decision needs to be performed subjectively by a human.

For the target phrase "The wind was so strong" examples of the chosen semantically similar sentences include the following: "It's surely a terrible storm", "Outside the wind rattled the tiles of the roof", "All night it had been blowing and raining". The chosen sentences for "It was the worst Sunday" involve: "It was a horrible journey", "The accident in question occurred upon the Sunday.", "He fell off dead". And the chosen ground truth segments for "I had much pleasure in reading": "I had the pleasure of meeting him in society", "Get in its favorite newspapers", "She enjoyed every hour of life".

[6]Cambridge Thesaurus published by Cambridge University Press https://tinyurl.com/y9c4yly4

Chapter 7

Baselines

In this chapter, we describe proposed approaches and tools for a baseline system for keyword search using semantic vectors.

7.1 Used tools

This thesis utilised standard Python machine learning frameworks including PyTorch, Scipy, ScikitLearn, Numpy, and Hugging Face. Data loading and processing were done by libraries such as Pandas, Librosa, and Rapidfuzz.

The computational work was conducted on a variety of hardware like Phonexia's Sun Grid Engine (SGE) clusters, a personal laptop and High-Performance Computing (HPC) systems, including LUMI and IT4I Karolina.

7.2 Proposed Search Algorithm

The proposed search algorithm consists of several steps: pre-processing of the input data, specifying the list of queries, and the search itself. The proposed workflow is presented in Figure 7.1.

The proposed search algorithm is explained in the following steps:

1. Pre-processing of the input dataset. This involves embedding extraction, model-specific data cleaning and normalisation.

   The data pre-processing includes audio resampling or cleaning of the input text by removing incompatible characters or splitting the audio/text into smaller chunks if it exceeds the maximum model context.

   The embedding extraction process is preceded by modality-specific actions. For text it is an input tokenisation, for audio it is a feature extraction. Afterwards, the embeddings are extracted by the model's encoder. Then the embeddings are saved together with the metadata, like the source path and the extracted segment timestamps.

2. Processing input queries. Queries can be entered either via text or via audio sample. Then each query is processed according to the modality the same way as in step `1.`. Thus, for each query, an embedding is obtained by the same model as for input data.

3. Pre-search processing. Each query can have a different length, it is therefore normalised by the *Frobenius norm* [17] or by 2-norm if the embedding is just one vector.

Frobenius norm is given by:

$$\|A\|_F = \left[\sum_{i,j} a \; bs \left(a_{i,j}\right)^2 \right]^{1/2} . \tag{7.1}$$

Then pooling is applied to the normalised vector sequence. The used pooling strategy is described in paragraph 7.3.

4. Search – vector comparison. First, a search window is set to the original size of the currently processed query. Then each embedding from the input dataset is compared using this sliding search to the query. The search window is sliding within the embedding with a certain hop. The embedding slice under the search window is pre-processed the same way as in the step 3.

   Vectors are compared either by cosine similarity – CS or by Euclidean distance[1] – ED. Thence, for each step of the sliding search window across the embeddings a similarity score (CD) or distance (ED) is obtained. Finally, slices of the embeddings where the scores are above (CS) or below (ED) a certain threshold are identified as the final search results. This step is performed for every input query.

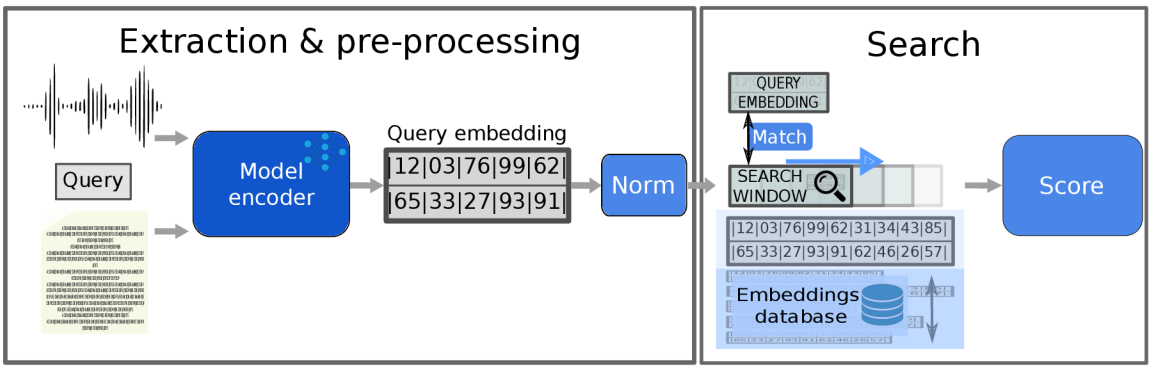


Figure 7.1: Workflow of the proposed search algorithm.

The proposed solution shows the search results with the following output format:

`["<"|">"]"THRESH":<FLOAT> ["COS"|"EUC"] (<INT>, <INT>) <FLOAT>":" <STRING>`

Individual parts of the output are explained as follows:

- Threshold. It utilises the following format: `["<"|">"]"THRESH":<FLOAT>`, where `<FLOAT>` represents any valid float number. The inequality symbols indicate whether the search results are above or below the threshold.
- Used method for vector comparison. `"COS"` represents the cosine similarity. `"EUC"` represents Euclidean distance.
- Position of the match (optional) – interval of the query found in the given data.
- Obtained score. The computed score for the given query.
- Text with the highlighted matched parts.

The examples of the output are shown in the following sections or in Appendix A.

[1] Also known as L^2 distance.

7.3 LaBSE baseline

This baseline is not multimodal as LaBSE works only with text input as explained in section 2.3.2. Nevertheless, LaBSE is a decent option for setting a baseline for the given task at least in the text domain. LaBSE gives decent results for multilingual tasks which is useful for text search as well. The model is trained to understand relations between sentences.

For this baseline, a pretrained model of LaBSE[2] is utilised. The search algorithm from section 7.2 was used as follows:

Sliding window step The sliding window step is set to 1 – as the used LaBSE model can provide embeddings tokenised per word. Therefore, it compares the query consecutively to the embeddings word by word. This property solves issues such as wrong comparison alignment. This way we omit meaningless comparisons. Those would occur if LaBSE produces embeddings per-character[3].

Processing of queries LaBSE adds start and end tokens for each sequence, which are then projected to the extracted embeddings as well. Therefore, these start and end tokens are cut from the extracted query off before comparison. Otherwise, this would cause the query to be more likely found at the beginning or the end of the compared sequences. The probability of being found in the middle of the sequences would be decreased.

The search window size The search window is given by the query size: $|win| = |query|$. However, for a one-word query, it would mean $|win| = |query| = 1$ which will lead to word comparisons only. Therefore, to see a wider context, the window is enlarged by the following equation: $|win| = |query| + \lfloor \alpha \times |query| + 1/2 \rfloor$, where $\alpha = 1/2$. The different search results for enlarged window size and without enlargement are shown in Figure 7.2. The results indicate significant a difference for Euclidean distance, whereas the difference for Cosine similarity is marginal. Since the results obtained by Euclidean distance show unstable behaviour, the Cosine similarity is used in the follow-up experiments.

Pooling strategies We compare two sequences of embeddings and we need to eventually compare two fixed-size vectors. This is even more important when we are changing the scope of the sliding window. Here, we have varying sizes of the embedding blocks for both the query and the input data. For pooling, some standard approaches like *mean* and *max* were utilised. Additionally, the *attention pooling* mechanism was used in the same way as proposed in Paragraph 7.4.1.

Example output from the search algorithm for the query "start" is presented in Figure 7.2. Some more detailed results for a longer query phrase are shown in Appendix A.1.

DET curves comparing the performance of the 3 search tasks are shown in Figure 7.3. The `Exact KWS` is the task with the evaluation set where only the literal query found is accepted, `KWS+SYN` is the task which extends the Exact KWS evaluation set by using synonyms of the query and all corresponding grammatical forms, and `SEM PHR` is the task

[2]LaBSE pretrained model by Google: https://tinyurl.com/249fle9x .

[3]For example, comparison of the query "Hello" to phrase "Welcome here!", with $step = 1$ will consecutively move window of $size = 5$ like the following: `[Welco]`, `[elcom]`, `[lcome]` and so on.

to match semantically close phrases where the evaluation set has manually selected phrases to match the target query phrase. The results show worse performance for `Exact KWS` and `KWS+SYN` tasks. After investigation, one of the reasons for the higher EER is an off-by-one alignment error, when sometimes the correct word is aligned to the ground truth word before the target one.

MinDCF eval metric is configured differently for each of the evaluation sets as the target prior probability differs for each approach ($p_{tar}^{exact} \approx 0.0001$, $p_{tar}^{semKWS} \approx 0.0005$). However, all share a higher cost for false rejection error of 10. Thus, MinDCF tolerates false acceptance errors more.

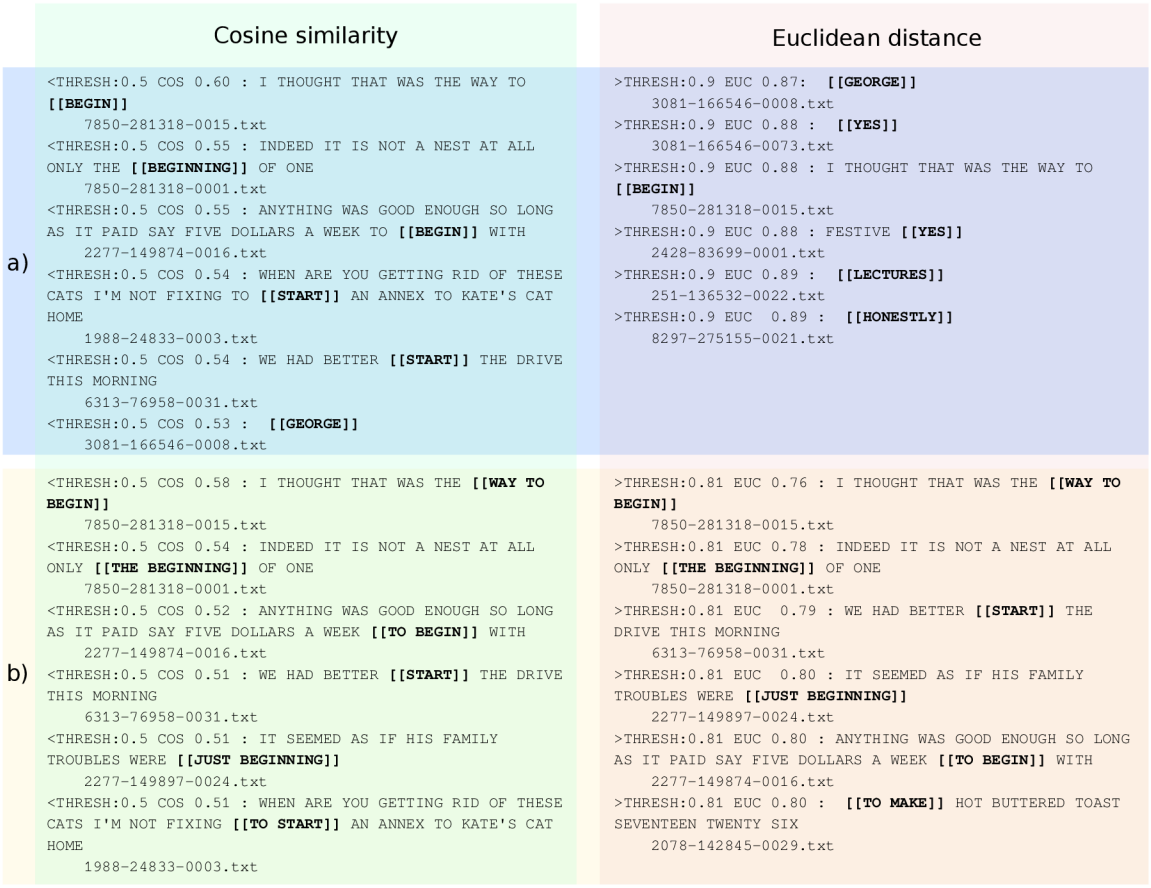


Figure 7.2: Samples of the search results of the LaBSE model. The prompted query is "start" and the used input text is the Librispeech dev subset. On the left side, the Cosine similarity for vector comparison is used, and on the right side the Euclidean distance. The search results in `a)` are obtained without extension of the search window and with extension `b)`.

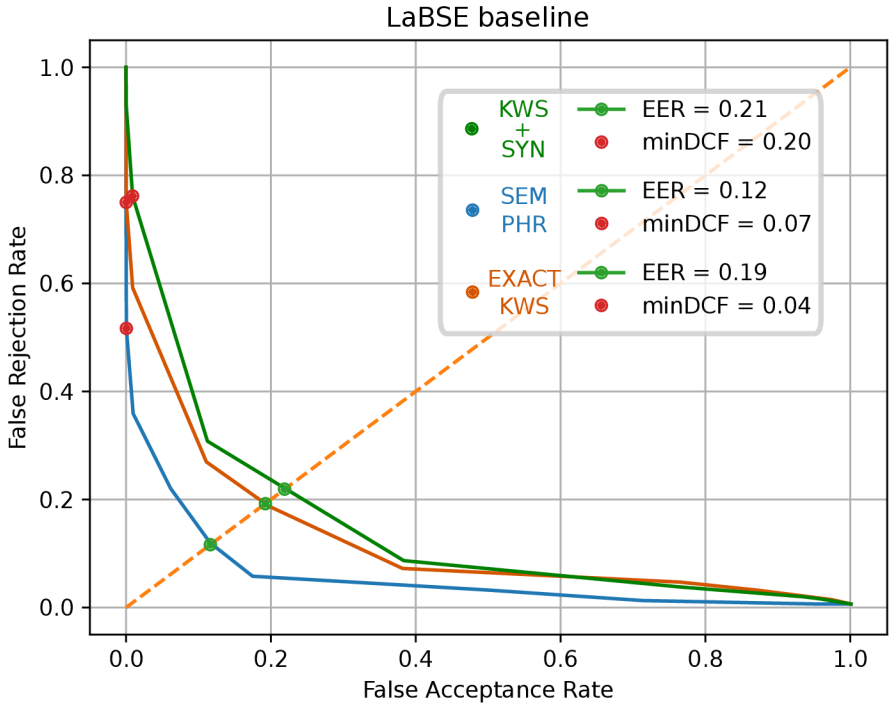


Figure 7.3: DET curve for three evaluation tasks. `Exact KWS` refers to the standard KWS task accepting only the exact query, `KWS+SYN` task extends the Exact KWS task with the query synonyms and all grammatical variations, and `SEM PHR` is the semantic phrases search task with evaluation set of manually chosen sentences that are semantically similar to the input query phrases. The evaluation sets are based on the Librispeech dev subset.

7.4 SpeechT5 baseline

The SpeechT5 model provides decent performance for the ASR, TTS and Speech-to-Speech (S2S) tasks. The JSALT 2023 workshop[4] at Le Mans Université showed that SpeechT5 is suitable also for intent classification task within research of Conversational models[5]. SpeechT5 finetuned to the intent classification task on the Slurp dataset[6] takes advantage of SpeechT5's text/audio multimodality support. The workshop's results show potential in the multimodal approach working with a shared vector space, where audio is used straightforward instead of an automatic transcription of speech to text mid step. This thesis builds upon these discoveries and aims to bring multimodal benefits for search in speech recordings regardless of the input. The initial overview of the quality of the shared vector space is shown in Figure 7.4.

The workflow of SpeechT5 stays in line with the proposed search algorithm and matches query embeddings to data as demonstrated in the LaBSE proof-of-concept. For the initial

[4] An intensive 6-week research workshop on speech and language engineering held in 2023 in Le Mans: https://www.clsp.jhu.edu/2023-jelinek-summer-workshop/

[5] Automatic design of conversational models from observation of human-to-human conversation: https://tinyurl.com/26ujuft7

[6] A Spoken Language Understanding Resource Package Creators (SLURP) – a challenging dataset in English spanning 18 domains available at Zenodo: https://zenodo.org/records/4274930

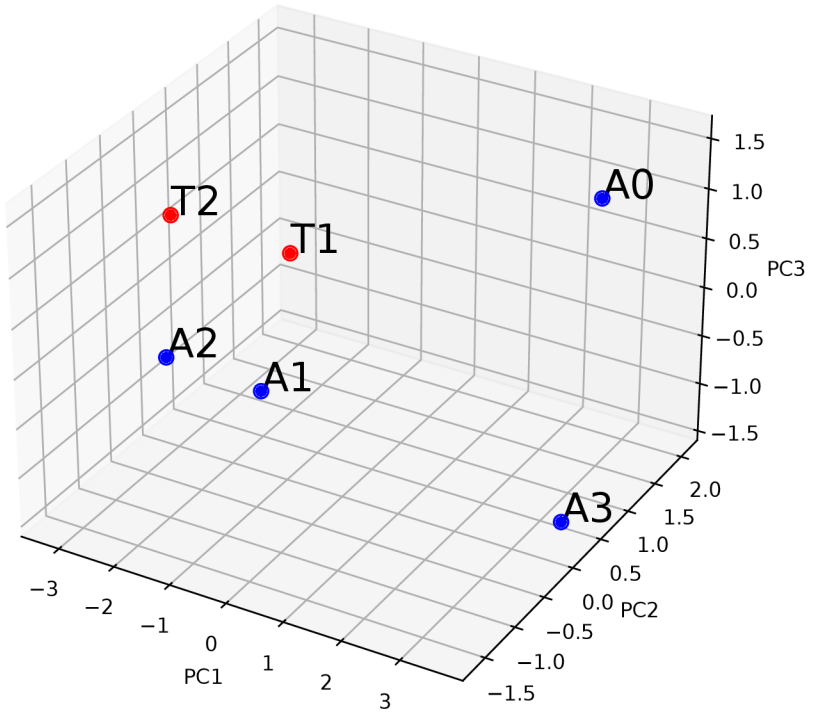


Figure 7.4: SpeechT5 shared vector space of six extracted embeddings reduced to the 3-dimensional space by principal component analysis. The red points `T2`, `T1` are text vectors and the blue points `A0`, `A1`, `A2`, `A3` are the audio vectors. Points `A1`,`T1` and `A2`,`T2` encodes datetime information, while `A0`, `A3` encodes completely unrelated utterance. From the plot, it is visible that vectors encoding similar information are closer regardless of the modality.

SpeechT5 baseline, a pretrained *base* model developed by Microsoft is utilised[7]. Since the other pretrained SpeechT5 models are available at Hugging Face[8], the base model is mapped to be compatible with the Hugging Face models. The SpeechT5 Hugging Face model was trained with ASR and TTS as objective tasks and outperformed the base model. These finetuned models perform better for the proposed search task as shown in the results.

7.4.1 Adaptation of the search algorithm

In this subsection, we will describe how the search using semantic vectors is utilised. Several setups were adopted from the LaBSE baseline. However, the search algorithm needs several adjustments due to the model complexity.

Approaches shared with LaBSE One of the shared approaches with LaBSE is omitting the first and the last token from the query. As it still helps to prevent a potential bias to prefer matching the beginnings and ends.

Extending the sliding search window size is adopted as well. However, SpeechT5 does not provide embeddings per word/bigger chunks but per frame/character. This raises an issue because the same phrase can be said faster or slower, that differs compared to searching purely over text. Therefore, the coefficient $\alpha = 1.0$. The reason to use this coefficient α

[7]SpeechT5 pretrained models repository by Microsoft: https://github.com/microsoft/SpeechT5
[8]Hugging Face SpeechT5 models: https://tinyurl.com/2d5pwn8u

value is to increase the chance of finding relevant results as Table 7.2 proves. To speed up the processing, the hop (step) is changed to $hop = 2$ for text and to $hop = 6$ for audio. Therefore, we are sliding the search window by 2 characters for text and by $96\,ms$ ($6 \times 16\,ms$) for audio.

Cross modality alignments SpeechT5 does not produce fixed-length embedding sequences across modalities. The default SpeechT5 embedding extracted from audio uses the default frame size $64\,ms$ and $16\,ms$ hop. For text, the size corresponds to the string length.

For example, an embedding for the word "start" (embedding size equals 5) is shorter than an embedding for audio containing spoken "start" (empirically, size can vary in an approximate interval between 25 and 45). Therefore, the search window should dynamically adjust when matching across modalities – *frame scaling*. This dynamic adjustment is accomplished by the properties saved in metadata during the extraction process. Therefore, with each embedding matched against the query, we scale the search window size based on the modality.

The scaler either widens or narrows the search window. From the obtained interval above – the comparison between string length and its audio embedding length, the scaler is set to 6. Therefore, when a text query is compared to an audio embedding, the search window is multiplied by 6. Conversely, when an audio query is matched to a text embedding, the search window frame is divided by 6. Otherwise, the window size remains unchanged.

It is important to note that generally, a wider search window yields more vague results, while a narrower window downgrades the semantic search capabilities, closer to the KWS task.

Pooling strategies An important part of the SpeechT5 baseline investigates the pooling strategies. As for LaBSE, the experiments started with standard mean and max pooling techniques. However, an additional technique was explored known as the *Self Attention Pooling* (SAP). SAP was implemented as proposed in Safari's Interspeech 2020 article [41]. SAP allows the model to focus on different parts of the input sequence while producing an output sequence, providing a kind of alignment between the input and the output. SAP is a method that uses self-attention mechanisms to aggregate dependencies between the input and the output sequence. This alignment between the input and output is particularly helpful for down-sampling pooling strategies.

SAP shows a better performance on the text embeddings than audio embeddings. Table 7.1 presents the performance of different pooling methods. Thus, for the proposed search algorithm, SAP is applied to text embeddings, and standard mean pooling is used for audio embeddings. The tests of pooling strategies are performed per modality. Audio queries are searched in audio embeddings and text queries are searched in text embeddings only.

Results The SpeechT5 baseline evaluation consists of measuring several search algorithm setups and testing the pretrained base model and finetuned Hugging Face models. The measurements were performed on the datasets derived from Librispeech dev on both text and audio. The systems were tested on tasks for `Exact KWS`, `Fuzzy KWS`[9], keyword synonym search (referred to as `KWS + synonyms`), and the `Semantic phrases search` (in tables abbreviated to *Sem phrases search*) task.

[9] `Fuzzy KWS` extends `exact KWS` task by allowing some Levenshtein distance between strings

The same input queries for the KWS + synonyms task were also used for Exact KWS and Fuzzy KWS tasks so it is objectively comparable. All of the input queries were entered using both text and audio. The audio queries were newly recorded using the laptop's microphone and Audacity[10]. The Cosine similarity is chosen as the vector comparison technique as Euclidean distance provides unstable results as can be seen in the example results from the LaBSE baseline.

The results in Table 7.1 and Table 7.2, and DET curves in Figure 7.5 and 7.6 show potentials of this solution. The best overall accuracy is obtained by the Hugging Face model with $collar = 2.0$. However, this collar setting makes the search window very wide as it is 3 times as large as the input query. The example outputs in the Appendix A.2, are subjectively considered to be general and semantically distant. Thus, the collar setting is chosen to $collar = 1.0$ as it is the best compromise between measured values and subjective evaluation.

Pooling	**Exact KWS** (text only) ↓EER	**Fuzzy KWS** (text only) ↓EER	**Exact KWS** (audio only) ↓EER	**Fuzzy KWS** (audio only) ↓EER
HF max	29.74 %	19.80 %	29.54 %	29.99 %
HF mean	0.21 %	13.54 %	**13.47** %	**13.70** %
HF attention	**0.14** %	**13.31** %	13.57 %	13.83 %

Table 7.1: Evaluation of different pooling strategies performed with the Hugging Face SpeechT5 model. Evaluated on Librispeech dev subset, for each modality separately. The figures show that the SAP technique performs better on text than on audio recordings. Though, the differences are minimal and they perform quite similarly.

Eval set	**Exact KWS**		**Fuzzy KWS**		**KWS + synonyms**		**Sem phrases search**	
Collar	↓EER %	↓MinDCF	↓EER %	↓MinDCF	↓EER %	↓MinDCF	↓EER %	↓MinDCF
Base 0.5	18.67 %	**0.26**	19.19 %	**0.11**	17.48 %	**0.14**	**14.03 %**	**0.06**
Base 1.0	17.63 %	0.27	18.09 %	0.11	16.58 %	0.15	14.16 %	0.07
Base 1.5	16.90 %	0.29	17.31 %	0.12	16.01 %	0.15	14.24 %	0.08
Base 2.0	**16.59 %**	0.30	**16.82 %**	0.13	**15.44 %**	0.16	14.65 %	0.09
HF 0.5	13.21 %	**0.21**	16.14 %	**0.11**	14.62 %	**0.13**	13.07 %	**0.06**
HF 1.0	12.24 %	0.22	15.03 %	0.11	13.61 %	0.14	13.43 %	0.06
HF 1.5	11.64 %	0.28	14.40 %	0.12	13.58 %	0.15	13.60 %	0.08
HF 2.0	**11.52 %**	0.30	**14.01 %**	0.13	**13.19 %**	0.16	**12.93 %**	0.09

Table 7.2: Collar settings for the base and Hugging Face SpeechT5 pretrained models. Evaluated on Librispeech dev subset according to the task, using the input queries as specified in Chapter 6. The chosen value for the collar is 1.0 as it is the best compromise between EER and MinDCF.

[10] Audacity https://www.audacityteam.org/

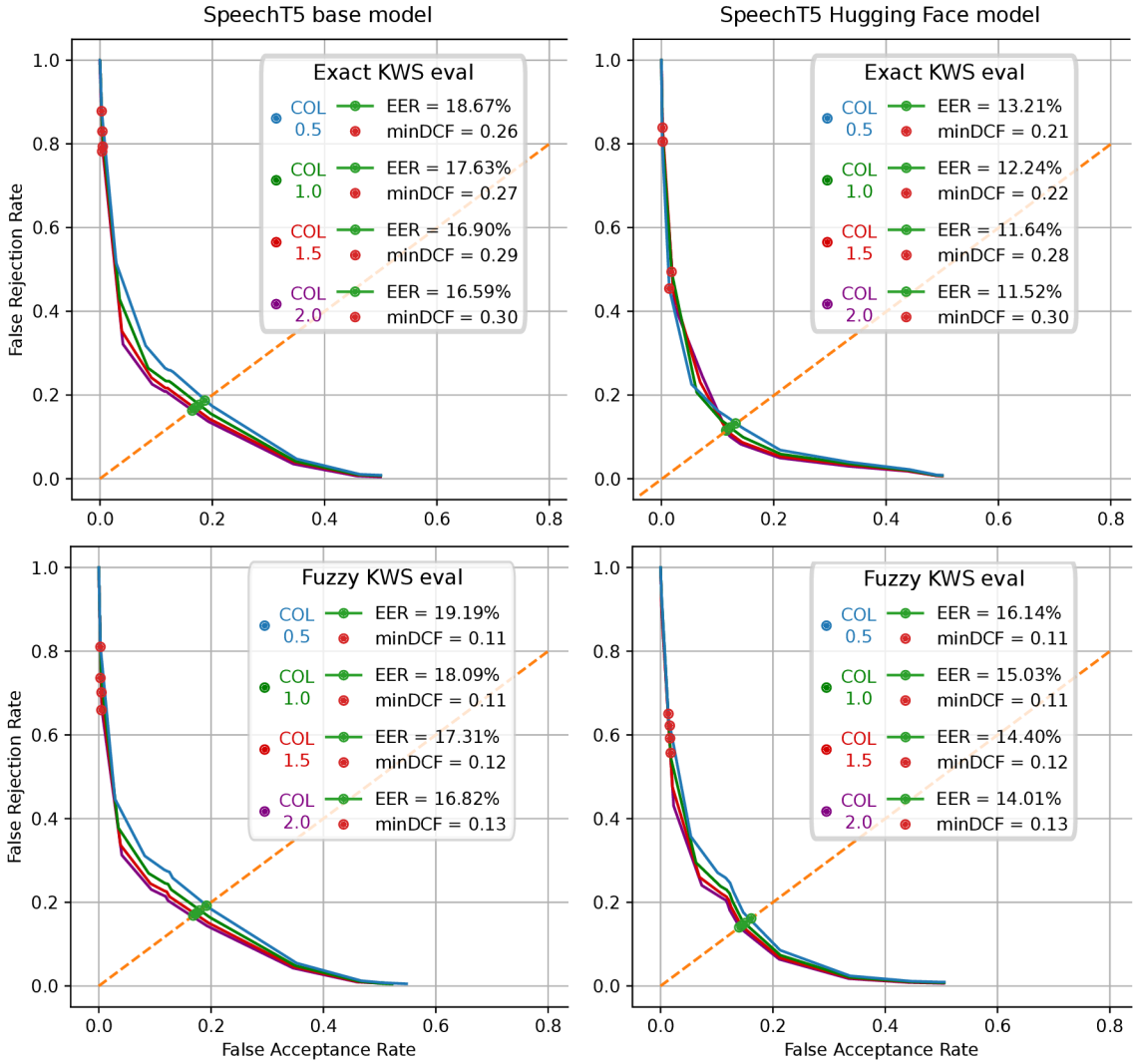


Figure 7.5: A comparison of the base and Hugging Face SpeechT5 pretrained models, with varying collar settings, performed on the keyword spotting (KWS) task. The DET curves for the base model are shown on the left and those for the Hugging Face model on the right. The Exact KWS only recognises the literal form of the query, whereas the Fuzzy KWS accepts close terms, measured by the Levenshtein distance. The evaluation sets are derived from the Librispeech dev subset.

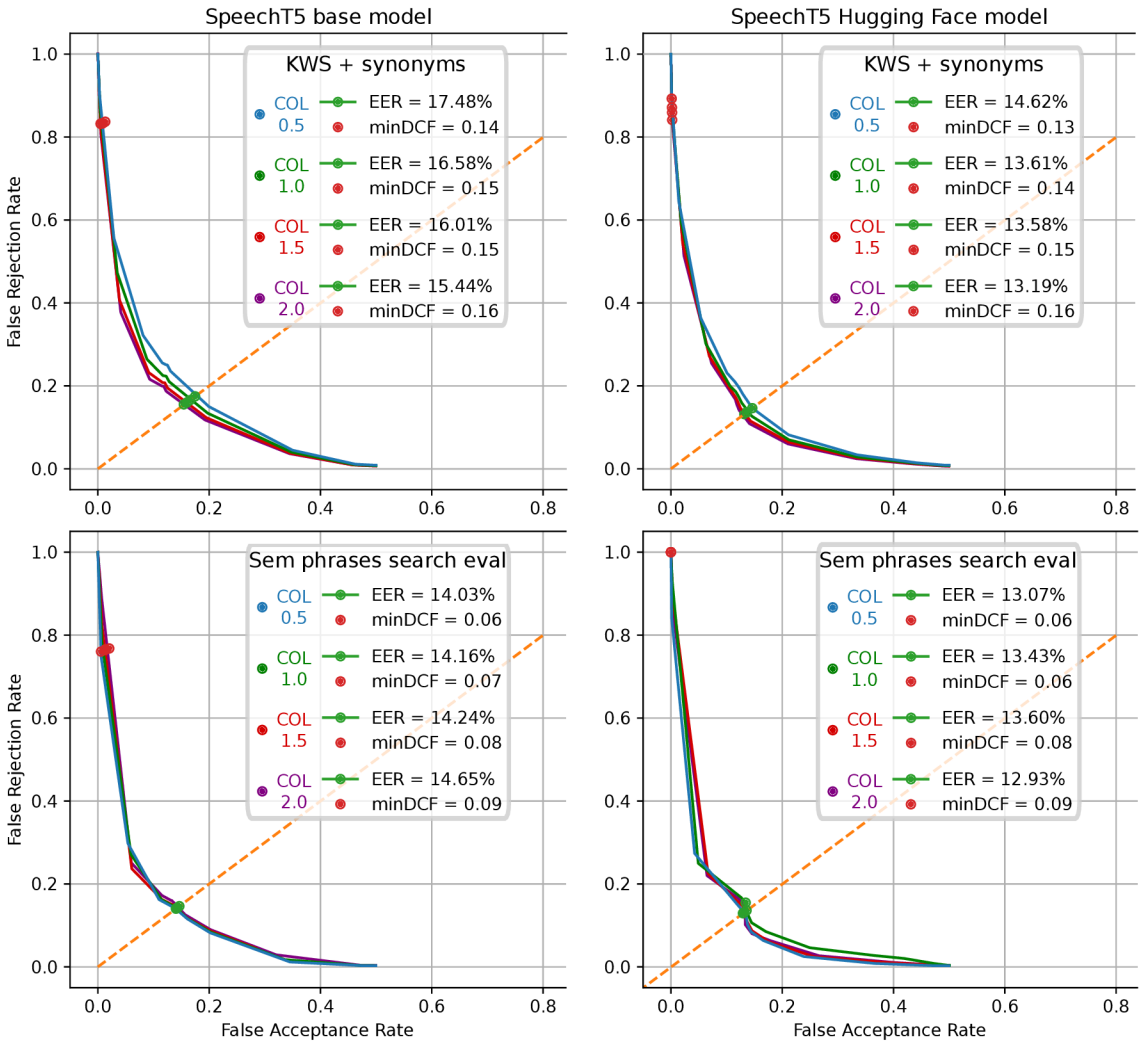


Figure 7.6: DET curves comparing the base and Hugging Face SpeechT5 pretrained models, with different collar settings, evaluated on sets for keyword spotting with synonyms (KWS + synonyms) and Semantic phrases search (Sem phrases search) tasks. The KWS with synonyms task evaluates matches for the query, its synonyms, and various grammatical forms. The Semantic phrases search task identifies sentences that are semantically similar to the query. The evaluation sets are derived from the Librispeech dev subset.

Chapter 8

Experiments

This chapter presents experiments and their results aimed at the enhancement of search accuracy and quality. We introduce experiments which investigate the capabilities of SpeechT5 model finetuning. Additionally, the SeamlessM4T model was tested to assess its advantages and disadvantages within the search. Finally, a new approach incorporating TTS systems into the search algorithm pipeline was explored.

8.1 Finetuning SpeechT5 models

In this section, we present an approach for finetuning SpeechT5 models to boost the search accuracy.

The main idea is to take Hugging Face pretrained models and learn them to focus more on the desired search window by limiting the context seen during training. We are limiting the context by feeding n-grams of text and the corresponding slices of audio during finetuning. To see the real impact of such an assumption, the finetuned models were trained from 3-grams segments, up to 11-grams, with a step-by-2 per finetuned model. The finetuning was performed on Hugging Face pretrained models finetuned for ASR tasks and TTS tasks. In total, 10 models were finetuned, 5 models per one Hugging Face pretrained model.

Data preparation First, several finetuning datasets were prepared. We used the Librispeech `train-clean-360` subset as the raw data. The dataset shares the structure of a typical ASR training dataset – audio recordings together with corresponding transcripts. The final format is compatible with the Hugging Face dataset format. The data were prepared as follows:

- Specified N-grams. The datasets were prepared for each experiment. Thus, for each 3/5/7/9/11-gram experiment, a new subset was prepared.
- N-gram segmentation from STM file. STM was created from Librispeech subset `train-clean-360` by forced alignment technique with each word (keyword) segmented separately to a new line. The N-gram segmented recording was created by selecting N segments (STM file rows) consecutively segment-by-segment (row-by-row) from the STM file. From these N-segments, the actual slice of the original recording was saved, together with the corresponding transcript – N words. For example, 5-gram recordings from the sentence "Nice day for fishing ain't it?" are created as

follows: `[Nice day for fishing ain't]` and `[day for fishing ain't it]`. Two recordings are created.

- Final count. Each subset consists of 10,000 N-grammed recordings with corresponding transcripts. The process from the previous bullet is repeated until the count is reached. This approach implies that the 11-gram dataset has more hours of recordings and therefore takes up more space than the 3-gram dataset to store.

Training The finetuning is performed with the ASR objective, thus we feed SpeechT5 with audio and use tokenised transcripts as the desired output. We run finetuning for 10 epochs (10,000 steps) with the Adam optimiser. Loss functions are used as proposed in the SpeechT5 paper during the finetuning. The finetuning is handled by the Hugging Face trainer.

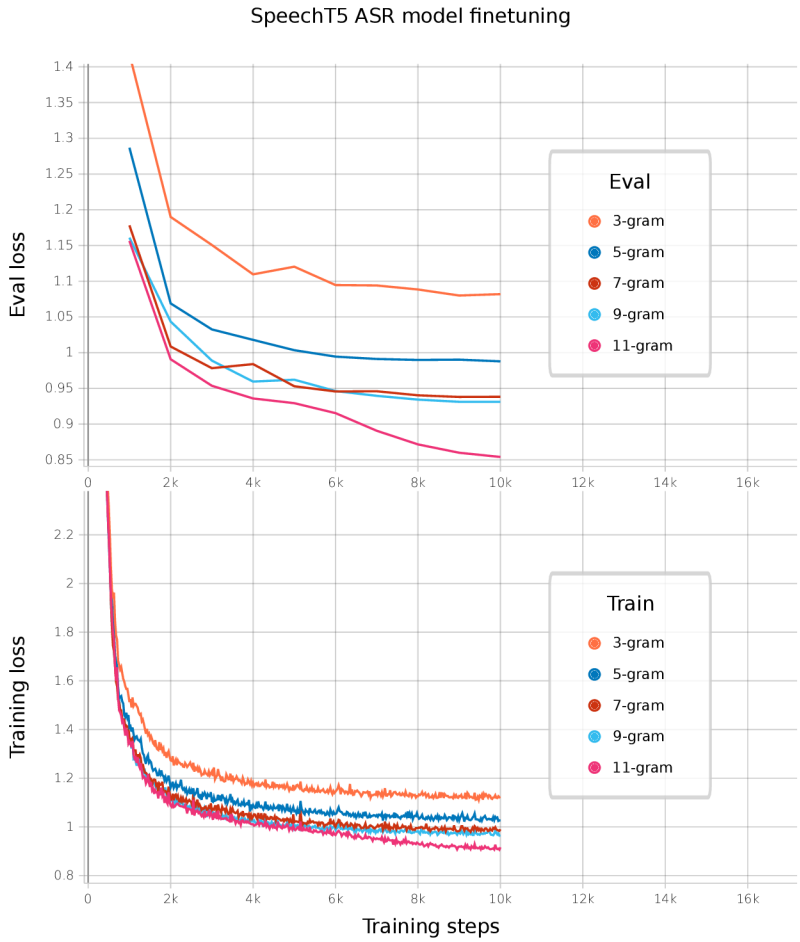


Figure 8.1: The finetuning process of the SpeechT5 ASR model involves various sets that limit the visible context. Figures show a noticeable difference between the training of the 3-gram and 11-gram approaches.

The initial learning rate is set to 10^{-5} with 500 warmup steps. For optimisation of performance, the gradient accumulation[1] step is set to 2.

The training process of the SpeechT5 ASR model (pretrained for the automatic speech recognition task) is shown in Figure 8.1 and for SpeechT5 TTS model (pretrained for the text-to-speech task) is in Figure 8.2. The finetuning was converging towards the objective. However, noticeably worse for the 3-gram approach, which could be expected as just 3 words are seen.

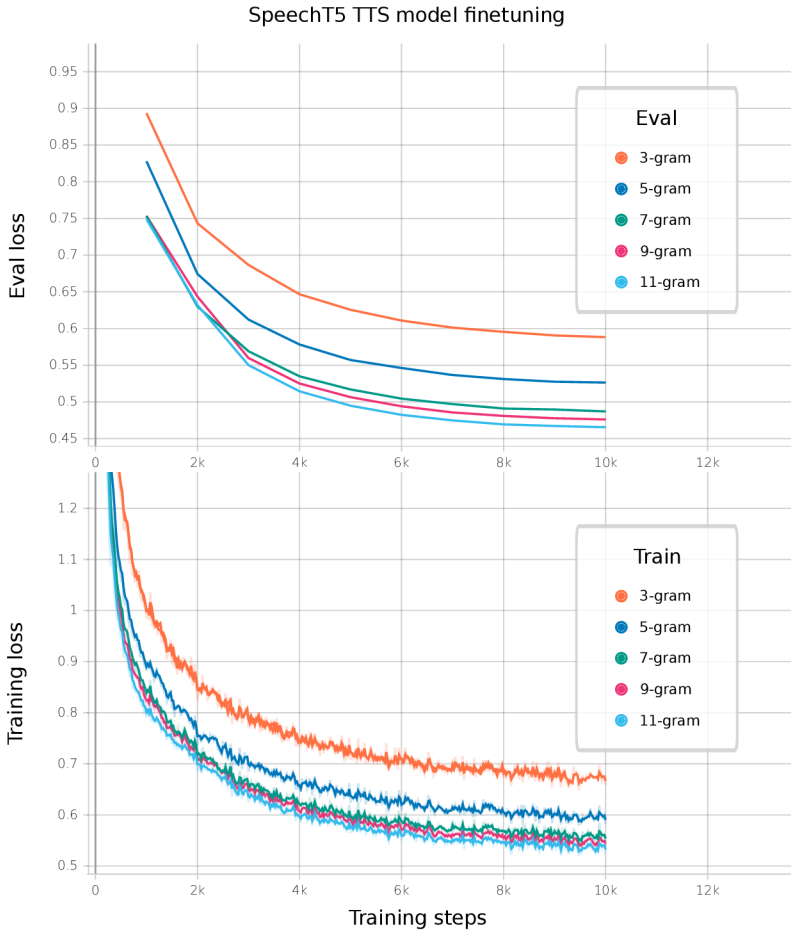


Figure 8.2: The finetuning process of the SpeechT5 TTS model with sets varying the visible context. Similarly to the finetuning of the SpeechT5 ASR model, the figures indicate a higher deviation for the 3-gram training process.

Results The primary objective of the proposed finetuning experiments was to develop a model dedicated to the keyword spotting task, or more generally, a model optimised for shorter queries in semantic search.

However, as shown in Table 8.1, limiting the context seen during finetuning does not make significant performance changes. Interestingly, TTS finetuned models performed

[1]Gradient accumulation is a strategy that enables training with larger batch sizes than hardware could typically fit in memory. It involves collecting gradients across multiple batches and updating the optimiser only after predefined steps of batches have been processed.

counter-intuitively better for 3-gram nearly every time – even for semantic phrases search. ASR finetuned models behaved more predictably. The 7-gram model performed better for Exact KWS and KWS + synonyms, indicating that a context of 7 words provides better embeddings. The 3-gram model performs the best at Fuzzy KWS as it matches keywords that are not just semantically similar, but also look/sound similar, thus it does not require a long context. The 11-gram ASR model performed the best for the semantic phrases search, proving that a wider context is beneficial.

DET curves in Figure 8.3 and Figure 8.4 show that the proposed approach for ASR and TTS models' finetuning, outperforms the chosen baseline Hugging Face model only for semantic phrases task, otherwise, it leads to worse performance.

Eval set	**Exact KWS**		**Fuzzy KWS**		**KWS + synonyms**		**Sem phrases search**	
Model	↓EER %	↓MinDCF	↓EER %	↓MinDCF	↓EER %	↓MinDCF	↓EER %	↓MinDCF
ASR 3	15.85 %	0.02	**13.01 %**	**0.09**	15.01 %	0.11	12.04 %	0.18
ASR 5	15.74 %	0.02	14.47 %	0.09	15.13 %	0.11	11.47 %	0.07
ASR 7	**12.72 %**	**0.02**	13.87 %	0.09	**14.59 %**	**0.11**	10.35 %	**0.04**
ASR 9	14.63 %	0.02	16.02 %	0.09	15.16 %	0.11	10.68 %	0.05
ASR 11	15.56 %	0.02	15.54 %	0.09	15.23 %	0.11	**10.19 %**	0.05
TTS 3	**14.12 %**	**0.02**	15.55 %	**0.09**	**14.90 %**	**0.11**	**11.17 %**	**0.04**
TTS 5	15.14 %	0.02	**15.15 %**	0.09	14.91 %	0.11	12.18 %	0.04
TTS 7	14.96 %	0.02	15.43 %	0.09	15.00 %	0.11	11.31 %	0.04
TTS 9	15.73 %	0.02	15.44 %	0.09	15.19 %	0.11	11.58 %	0.04
TTS 11	16.05 %	0.02	15.67 %	0.09	15.42 %	0.11	12.45 %	0.04

Table 8.1: The evaluation of SpeechT5 finetuning experiments. The results show that models finetuned on the ASR model provide better and more predictable outcomes.

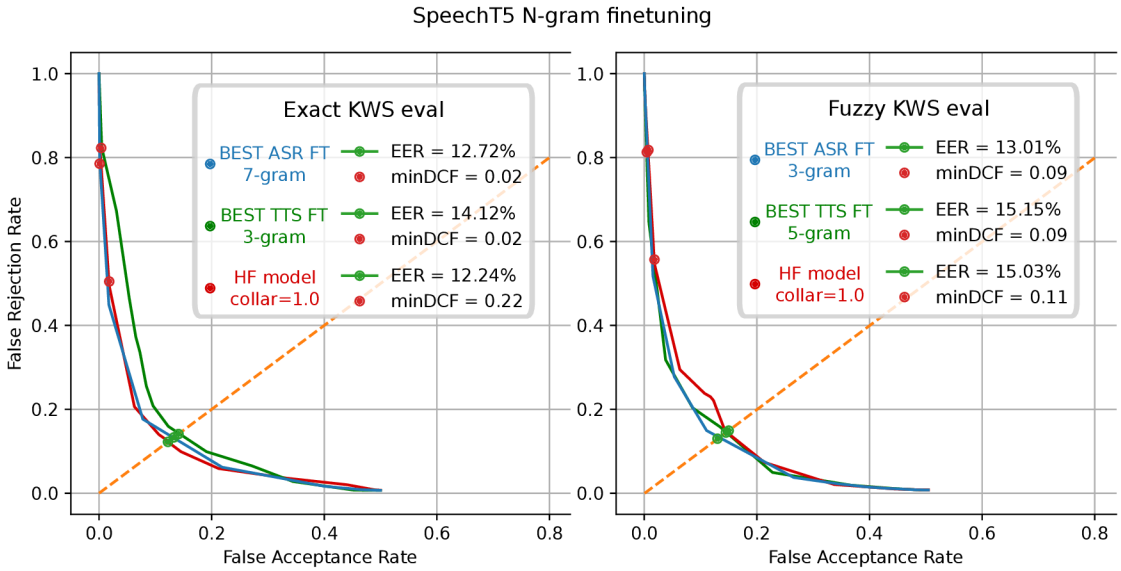


Figure 8.3: DET curves of the best finetuned ASR model and TTS model compared to the chosen baseline Hugging Face model. Evaluated on sets for the Exact KWS task (left) and the Fuzzy KWS task (right) .

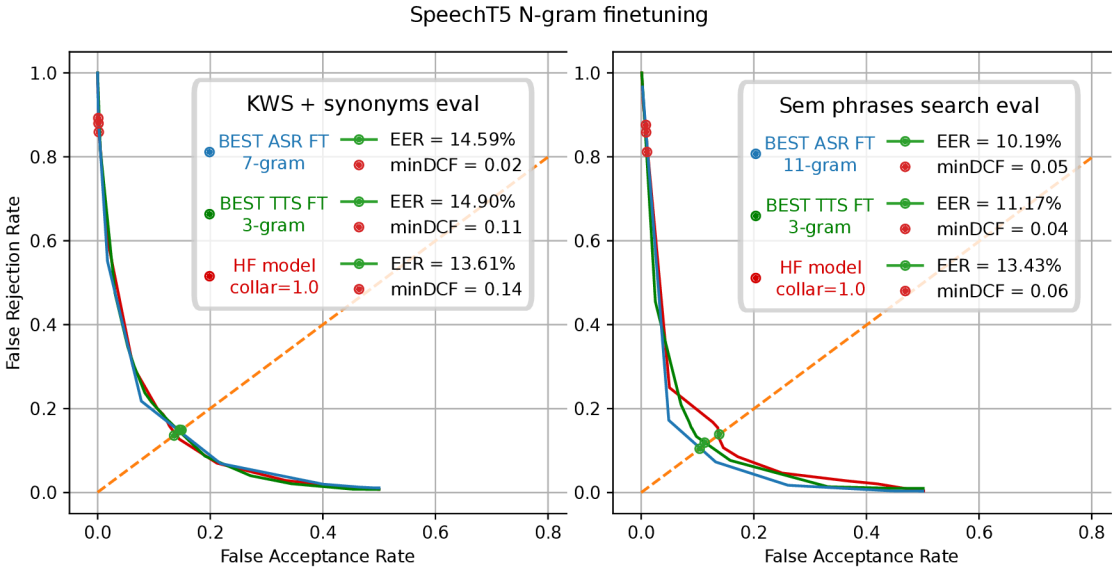


Figure 8.4: DET curves of the best finetuned ASR model and TTS model compared to the chosen baseline Hugging Face model. Evaluated on sets for the KWS + synonyms task, and the Semantic phrases search task.

8.2 SeamlessM4T model

In this section, experiments with the SeamlessM4T model are presented.

SeamlessM4T is a robust multi-modal multi-lingual model. It should offer better joint vector space – SONAR, as it was trained on more diverse data and utilises advanced alignment techniques as explained in section 4.2.

The model also provides more advanced approaches which solve some of the issues of the SpeechT5 model. The main improvement is the alignment between audio and text embeddings. As a result, the embeddings are segmented to a size which corresponds to the length of a character for both text and speech.

SeamlessM4T also allows us to put longer text sequences at the input than SpeechT5. SpeechT5 allows a maximum of 450 tokens for text and 4000 frames for audio (corresponds to 64 seconds for default settings of SpeechT5 feature extractor), while SeamlessM4T v2 allows for 2048 tokens for text. The audio maximum length is similar allowing 4096 units.

Several pretrained models are available for SeamlessM4T[2]: *v2-Large*, *v1-Large*, *v1-Medium*, *v1-Unity-Small*. For the experiments, we used `SeamlessM4T v2`[3] pretrained model from Hugging Face.

However, the model is 8 times larger than the SpeechT5 models and thus requires a larger GPU and longer processing time.

Thanks to the improvements, the search algorithm workflow is more similar to the LaBSE baseline as there is no need to use the so-called "scaler" when matching across modalities. However, SeamlessM4T uses the character tokenisation, not words and thus the collar is set to 1 which worked better for SpeechT5 models.

[2]SeamlessM4T pretrained models by Meta on Hugging Face: https://tinyurl.com/2b2fwm2k
[3]SeamlessM4T v2 pretrained model: https://tinyurl.com/2bgn5b42

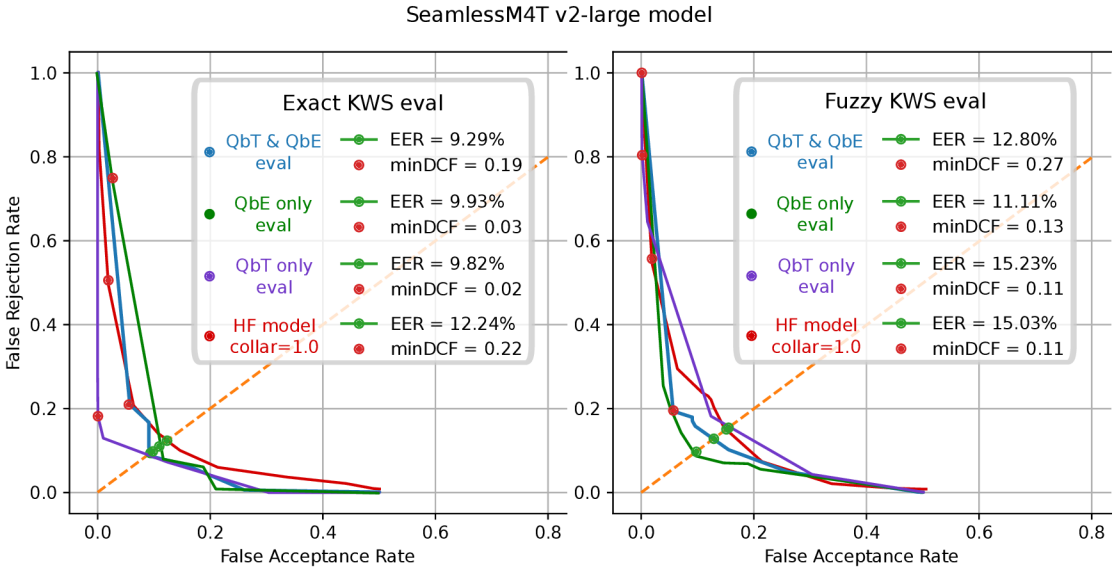


Figure 8.5: DET curves of the SeamlessM4T-based system and the SpeechT5 baseline system for exact KWS and fuzzy KWS tasks. The SeamlessM4T model outperforms the baseline SpeechT5 model (in red). The blue DET curve evaluated jointly on input queries entered by text (QbT) and by audio (QbE) shows elbow around the EER operation point. Additional DET curves (green, purple) evaluate QbT input and QbE separately. The SeamlessM4T model requires different calibrations depended on the input modality.

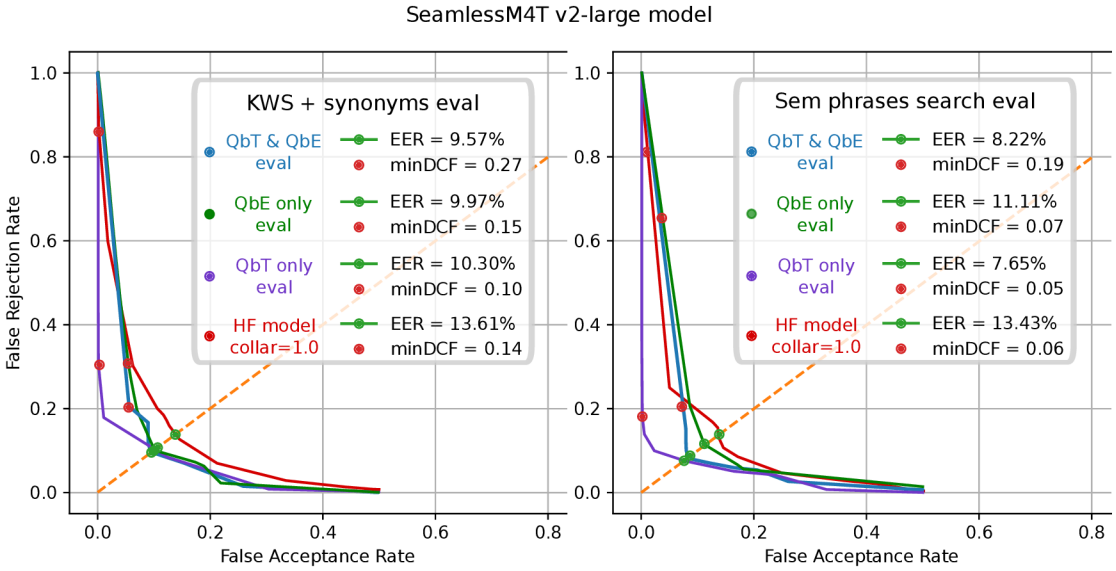


Figure 8.6: DET curves of the SeamlessM4T-based system and the SpeechT5 baseline system for KWS with synonyms and Semantic phrases search tasks. The SeamlessM4T model outperforms the baseline SpeechT5 model (in red). The elbow for joint QbT & QbE evaluation (blue) is present as well.

Results SeamlessM4T shows potentials in various ways. From the examples shown in Appendix A.3, it is visible that the quality of the SONAR joint embedding space is higher

Eval set	Exact KWS		Fuzzy KWS		KWS + synonyms		Sem phrases search	
Model	↓EER %	↓MinDCF	↓EER %	↓MinDCF	↓EER %	↓MinDCF	↓EER %	↓MinDCF
QbT & QbE	**9.29 %**	0.19	12.80 %	0.27	**9.57 %**	0.27	8.22 %	0.19
QbE only	9.93 %	0.03	**9.68 %**	0.13	9.97 %	0.15	11.11 %	0.07
QbT only	9.82 %	**0.02**	15.23 %	**0.11**	10.30 %	**0.10**	**7.65 %**	**0.05**

Table 8.2: The SeamlessM4T system's performance on the LibriSpeech dev-clean subsets is assessed using both audio query-by-example (QbE) and query-by-text (QbT). It is consistent with all other measurements. Additionally, the DET is evaluated separately for each input modality due to differing optimal operation points.

than the SpeechT5 embeddings. SeamlessM4T shows similar scores for matching the content either in text or audio. It may be caused by the better alignment between modalities. Also, the false alarms are more semantically close to the target query, than SpeechT5-based models.

These statements are supported by the evaluations, as seen in Table 8.2 and in DET curves 8.5 and 8.6.

Evaluation is performed in the same way as with SpeechT5 models. Both query-by-text (QbT) and query-by-example (QbE) are evaluated together. In DET curves a visible elbow appeared. Therefore, separate measurements for input query modalities were conducted. The modalities within subsets were left unchanged. SeamlessM4T seems to require different operation point calibrations depending on the modality of the input.

The reason for these evaluations was the visible elbow in the DET curve showing joint input modalities. This claim was supported by additional tests where such a significant jump no longer occurred.

The main downside of the SeamlessM4T model is the higher demands on computational resources and a longer time needed for processing, and therefore for searching. However, the capabilities of SeamlessM4T models are significantly broader due to the multilingual nature of the models. To test these capabilities, some more in-depth research is required, including the use of specified multilingual datasets labelled for the primary objective of this thesis.

8.3 TTS approach

In this section, we present a different approach to the workflow of the proposed search algorithm. The proposed experiments involve use of the text-to-speech systems to enhance the quality of the shared embedding space and therefore better performance of the system.

The idea is to add a TTS system before the embedding extraction step for text input. After the text tokenisation, the input ids are forwarded to TTS in order to generate speech. This speech output is then used for the embedding extraction. This approach ensures that similar embeddings are close together, regardless of the input modality, as the model only transforms speech into a joint vector space. The edited workflow can be seen in Figure 8.7. It remains unchanged for audio input.

This approach eliminates several issues, such as text/speech alignment, thus avoiding the need for additional steps like utilising a "scaler". It also leads to a more error-resistant system compared to the opposite approach of this idea – ASR system used to create a textual query from audio. ASR (or speech-to-text) systems are prone to errors in the transcription

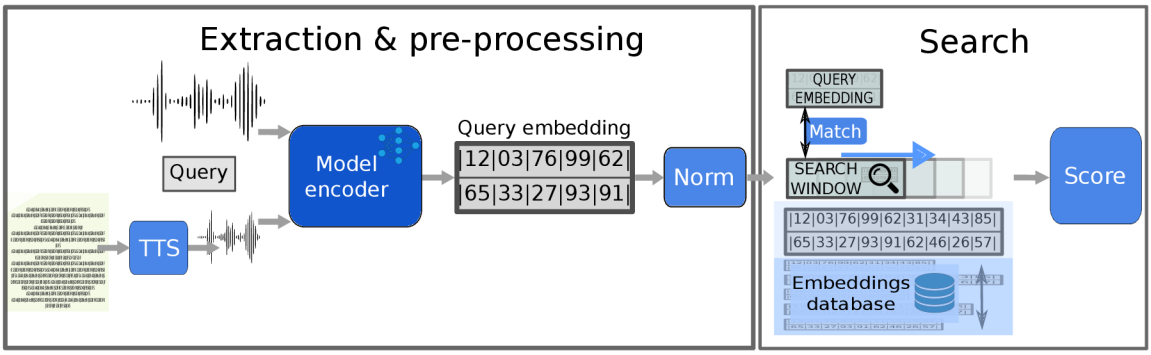


Figure 8.7: The edited workflow of the search algorithm to involve the TTS module.

of audio-to-text. Therefore, TTS systems offer a more reliable and robust solution, as text lacks the variations encoded in speech, such as pitch, speed, and environmental noise.

However, it comes with some drawbacks. The additional step in the workflow slows down the system and the use of TTS is a performance bottleneck in the chain. Furthermore, it introduces model-specific constraints, such as maximum input size, supported languages, or batch size.

In the TTS approach experiments, the SpeechT5 TTS and SeamlessM4T models are used. SpeechT5 provides a smaller model exclusively for English, with a batch size restricted to one. SpeechT5 requires a speaker embedding to generate speech. It supports 512-dim X-vectors. Some speaker embeddings from CMU ARCTIC dataset[4] were used. For all experiments, the same X-vector[5]

SeamlessM4T supports up to 35 languages for speech output and it supports batch processing for TTS. Speaker embeddings can be specified or remain at the default one preselected for the language. Default speaker embeddings are used throughout these experiments.

Results The SpeechT5 TTS approach demonstrates the best performance among systems based on SpeechT5. It also delivers decent results for the SeamlessM4T-based system. Nonetheless, the performance gap between the SeamlessM4T system described in section 8.2 and the SeamlessM4T with the TTS approach is minimal, with most of the evaluation sets showing slightly lower scores, except for the Semantic phrases search task. Some detailed results are presented in Table 8.3 and corresponding DET curves are presented in Figure 8.8 and Figure 8.9.

The TTS approach appears to assist SpeechT5 in resolving the alignment issue between modalities, subsequently demonstrating performance comparable to advanced architectures such as SeamlessM4T.

[4]X-vectors extracted from CMU ARTICS set at Hugging Face: https://tinyurl.com/27qryvcw

[5]Speaker embedding from CMU ARTICS dataset with index `7306`, which belongs to US English female voice: `cmu_us_slt_arctic-wav-arctic_a0506`

Eval set	Exact KWS		Fuzzy KWS		KWS + synonyms		Sem phrases search	
Model	↓EER %	↓MinDCF	↓EER %	↓MinDCF	↓EER %	↓MinDCF	↓EER %	↓MinDCF
SpeechT5 TTS approach	10.72 %	**0.03**	**10.00 %**	**0.11**	14.29 %	**0.13**	10.16 %	**0.06**
SeamlessM4T TTS approach	**10.04 %**	0.04	10.42 %	0.16	**10.27 %**	0.19	**7.48 %**	0.08

Table 8.3: TTS approach systems results show similar performance for Exact and Fuzzy KWS tasks regardless of the used model. For the KWS + synonyms task and the Semantic phrases search task, SeamlessM4T performs better.

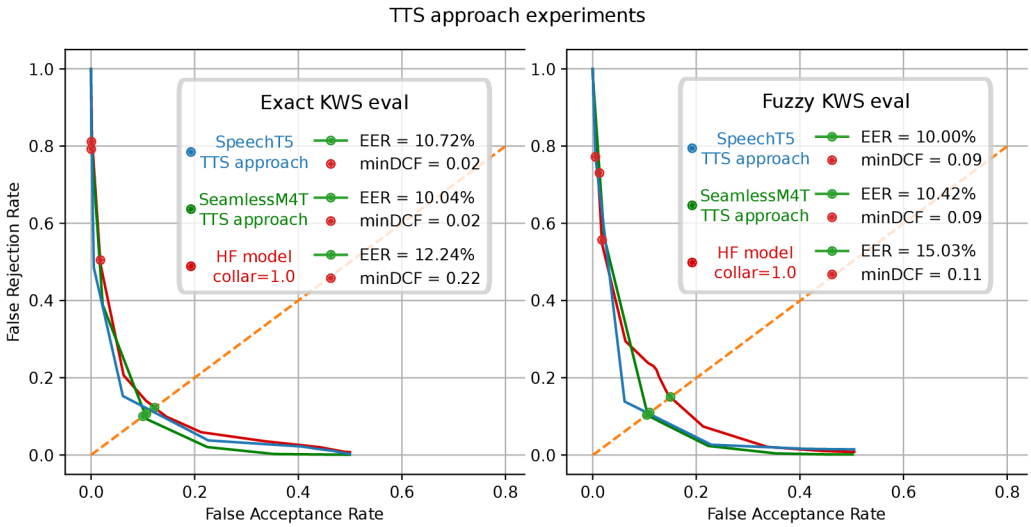


Figure 8.8: DET curves showing the performance of TTS approach systems compared to the SpeechT5 baseline, evaluated on sets for Exact KWS and Fuzzy KWS tasks.

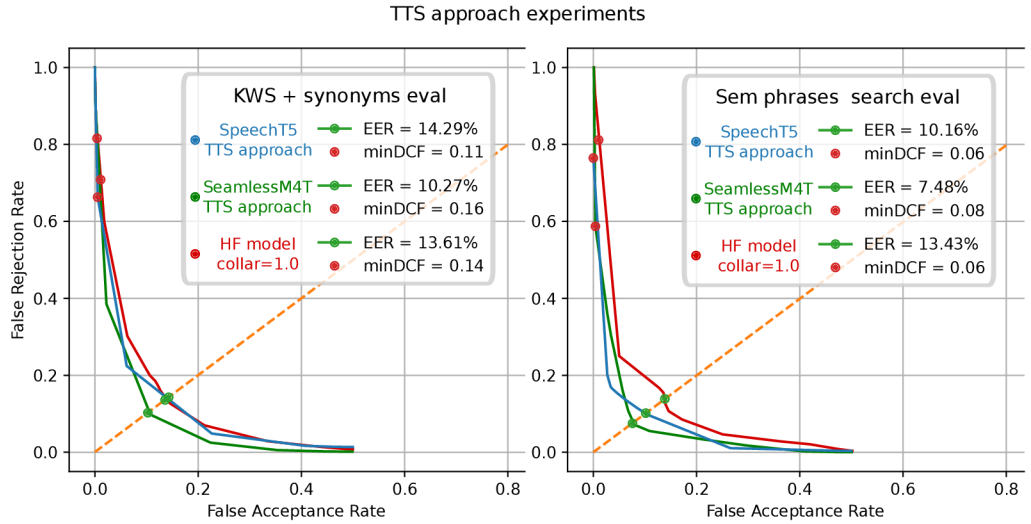


Figure 8.9: DET curves showing the performance of TTS approach systems compared to the SpeechT5 baseline, evaluated on sets for KWS + synonyms and Semantic phrases search tasks.

8.4 Final comparisons

This section presents and concludes overall comparisons among the top-performing systems. Additionally, it includes a comparison with the existing keyword spotting system developed by Phonexia.

Keyword spotting systems comparison Phonexia Keyword Spotting is used as a referential system to compare created systems. The evaluation tool for Phonexia KWS outputs the ROC curve, together with precision and recall of the operation point chosen by Phonexia. To compare the created systems, an operation point around the minDCF point is selected for estimating precision and recall. Measurements are performed on the Librispeech dataset. The results are presented in Table 8.4.

The ROC curves in Figure 8.10 indicate that legacy systems outperform proposed solutions, however, the differences are not significantly big. This gives a positive indication that the system, built for semantic search tasks mainly, can compete with the exact keyword spotting systems. In addition, created systems are multimodal and are capable of searching in both text and audio, while Phonexia's solution searches in audio only. Moreover, all newly investigated solutions support both QbT and QbE at the input.

KWS System	**↑ Precision**	**↑ Recall**	**↑ ROC AUC**
Phonexia KWS	98.59 %	99.61 %	0.9658
SpeechT5 HF (collar 1.0)	92.66 %	79.37 %	0.9432
SpeechT5 FT (ASR, 7-grams)	91.37 %	82.34 %	0.9430
SpeechT5 TTS approach	**93.22** %	84.72 %	0.9514
SeamlessM4T v2-Large	90.92 %	90.46 %	0.9218
SeamlessM4T TTS approach	89.36 %	**90.74** %	**0.9571**

Table 8.4: The table displays precision and recall metrics for systems evaluated for the exact KWS task, calibrated to operational points near minDCF. ROC Area Under Curve (ROC AUC) shows high numbers across all systems.

Fisher subset SpeechT5-based systems were also evaluated on the Fisher subset presented in section 5.2. Before processing, it was necessary to make several changes as explained in that section. Two approaches to create a mono channel recording from stereo recordings were evaluated: `i)` split stereo channels to two separate mono channel recordings (referred to as *Split2Mono*) and `ii)` merge stereo recording into one mono channel recording (referred to as *Merge2Mono*). This also creates an imbalance in the total speech length, as *Split2Mono* has two times more of the recordings.

Additionally, SpeechT5 is unable to process lengthy recordings, such as 10-minute conversations within the Fisher dataset. As a result, these recordings are divided into chunks of 34 seconds each. Each chunk consists of 30 seconds that do not overlap with adjacent chunks and with an additional 2-second overlap at the beginning and the end. Similarly, the

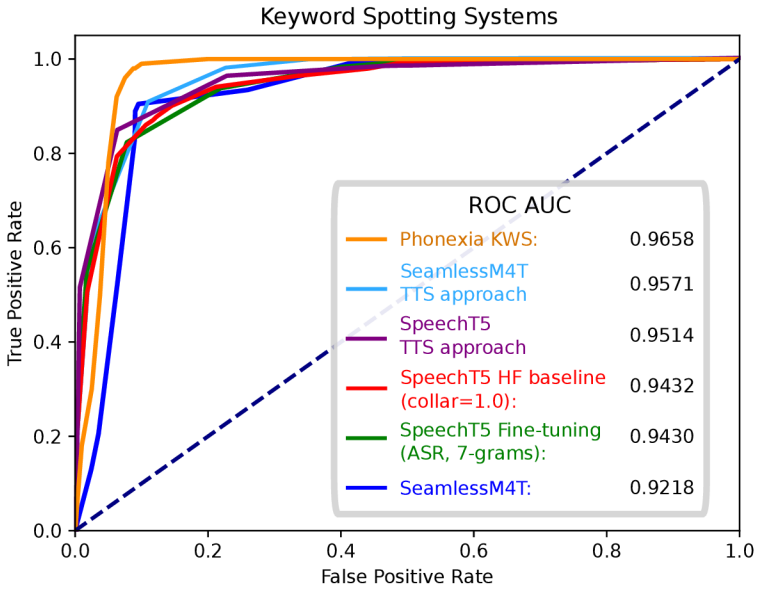


Figure 8.10: ROC curves comparing systems for the Exact KWS task. Phonexia KWS outperforms proposed systems. As Area Under Curve (AUC) indicates, TTS approach systems perform close to Phonexia KWS.

text must also be shortened for processing. It is divided into strings of 440 characters each, where 400 characters do not overlap with neighbouring strings, and there is a 20-character overlap at the beginning and the end.

The results are displayed solely for the Exact and Fuzzy KWS tasks, as the ground truths for these tasks are automatically estimated from the STM file, and the KWS with synonyms task and Semantic phrase search task require manual labelling. The DET curves are shown in Figure 8.11 and Figure 8.12. DET curves show a significant decrease in overall accuracy.

The drop was expected since the Fisher data are telephone conversations with natural speech imperfections. Also, compared to Librispeech, even chunked recordings are longer, which naturally results in more false acceptance errors. Upsampling of data to 16 kHz due to the SpeechT5 model can also worsen the result – as the model is trained on 16 kHz recordings.

The results indicate, that further development is required to meet the production level quality of the technology. Finding an optimal threshold for such systems can be challenging due to lower performance as it will either significantly increase the false acceptance rates or decrease the true acceptance rates.

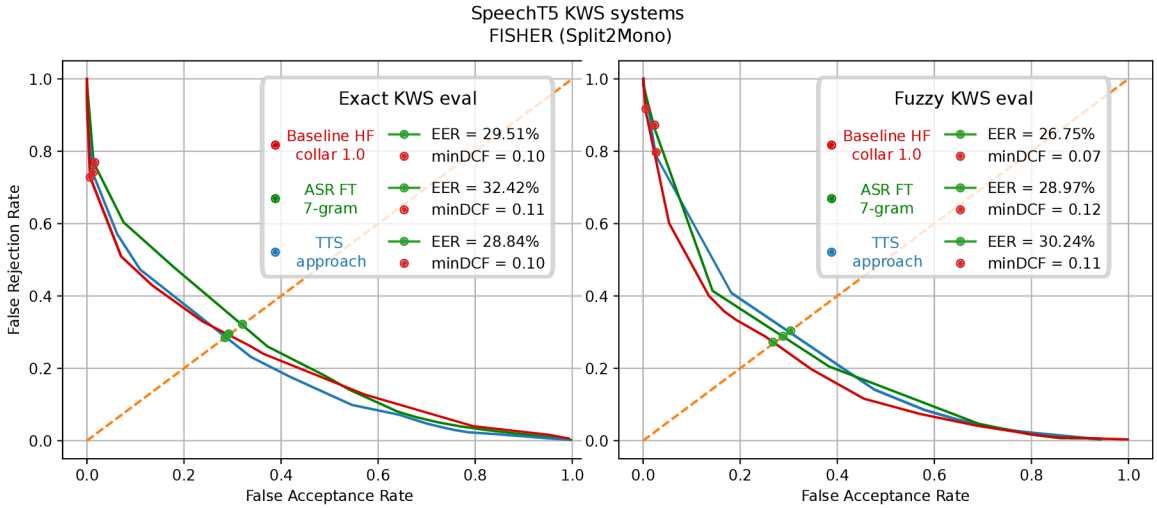


Figure 8.11: DET curves comparing the best performing SpeechT5 systems on Fisher dataset (Split2Mono conversion). DET curves show a significant decrease in overall accuracy.

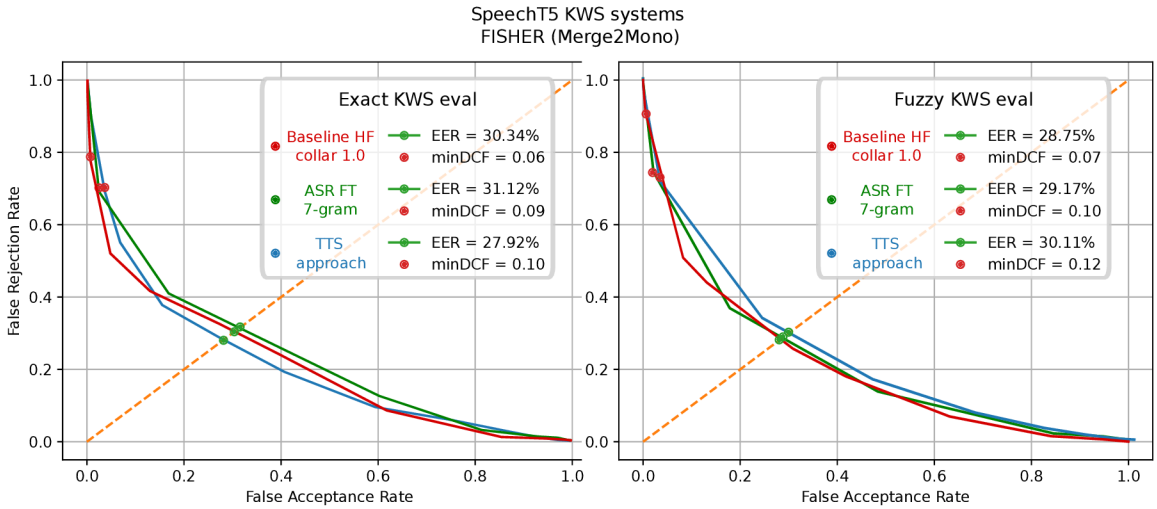


Figure 8.12: DET curves comparing the best performing SpeechT5 systems on Fisher dataset (Merge2Mono conversion).

Chapter 9

Conclusion

9.1 Summary of the work performed

In this thesis, we have presented the methods for using semantic vectors to search in speech. Particularly, we focused on joint representation models, which transform text and audio data into a shared vector space. With these multimodal models, we proposed the search algorithm where we tried to find optimal settings for the pretrained SpeechT5 models.

Additionally, we performed several experiments to improve the performance of the baseline. This included finetuning, where we focused on limiting the seen context and analysed its impact on performance. We experimented with a novel, more robust architecture of joint representation models – SeamlessM4T, and attempted to adjust the properties of the model to suit the search algorithm. At the end, we increased the accuracy of the proposed system by incorporating a text-to-speech (TTS) mechanism into the workflow.

In order to evaluate the systems for the semantic search tasks, we created labelled subsets using a thesaurus for the semantic keyword spotting task. Additionally, we manually labelled semantically close sentences for the semantic phrase search task.

We evaluated the solutions against each other using datasets for speech-to-text systems. Furthermore, we compared the best systems with a legacy commercial keyword spotting product.

The results show a high potential for this novel approach in audio search. However, some additional development is required for a commercial deployment. This primarily includes accuracy improvement, decreasing the false alarm rate, and increasing the speed of both embedding extraction and search.

9.2 Future work

Additional improvement could focus on investigating the use of the TTS system with models other than multimodal ones. The Wav2vec2 model and its modifications could serve as the way for further research.

Moreover, the SeamlessM4T model and its multilingual capabilities require further investigation. Such a cross-lingual system could assist a wide array of analytical professions.

For the deployment of the proposed system to production and to enable the processing of high loads, optimisations are required. The main bottleneck lies in the slower extraction of embeddings and the search process itself. For faster vector comparisons, the *Faiss* library developed by Meta can be utilised. This should significantly improve the performance.

Bibliography

[1] Alim, S. A. and Rashid, N. K. A. Some Commonly Used Speech Feature Extraction Algorithms. In: Lopez Ruiz, R., ed. *From Natural to Artificial Intelligence*. Rijeka: IntechOpen, 2018, chap. 1. DOI: 10.5772/intechopen.80419. Available at: https://doi.org/10.5772/intechopen.80419.

[2] Ao, J., Zhou, L., Wang, C., Ren, S., Wu, Y. et al. SpeechT5: Unified-Modal Encoder-Decoder Pre-Training for Spoken Language Processing. In: Muresan, S., Nakov, P. and Villavicencio, A., ed. *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, p. 5723–5738. DOI: 10.18653/v1/2022.acl-long.393. Available at: https://aclanthology.org/2022.acl-long.393.

[3] Baevski, A., Hsu, W.-N., Xu, Q., Babu, A., Gu, J. et al. Data2vec: A General Framework for Self-supervised Learning in Speech, Vision and Language. In: Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G. et al., ed. *Proceedings of the 39th International Conference on Machine Learning*. PMLR, 17–23 Jul 2022, vol. 162, p. 1298–1312. Proceedings of Machine Learning Research. Available at: https://proceedings.mlr.press/v162/baevski22a.html.

[4] Baevski, A., Zhou, H., Mohamed, A. and Auli, M. Wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2020. NIPS'20. ISBN 9781713829546.

[5] Chen, S., Wang, C., Chen, Z., Wu, Y., Liu, S. et al. WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing. *CoRR*. 2021, abs/2110.13900. Available at: https://arxiv.org/abs/2110.13900.

[6] Chen, Y.-C., Huang, S.-F., Lee, H.-y., Wang, Y.-H. and Shen, C.-H. Audio Word2vec: Sequence-to-Sequence Autoencoding for Unsupervised Learning of Audio Segmentation and Representation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*. 2019, vol. 27, no. 9, p. 1481–1493. DOI: 10.1109/TASLP.2019.2922832.

[7] Chung, Y.-A., Zhang, Y., Han, W., Chiu, C.-C., Qin, J. et al. W2v-BERT: Combining Contrastive Learning and Masked Language Modeling for Self-Supervised Speech Pre-Training. In: *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. 2021, p. 244–250. DOI: 10.1109/ASRU51503.2021.9688253.

[8] Cieri, C., Graff, D., Kimball, O., Miller, D. and Walker, K. *Fisher English Training Speech Part 1 Speech* [Web Download]. Philadelphia: Linguistic Data Consortium, 2004. DOI: 10.35111/da4a-se30. LDC Catalog No. LDC2004S13.

[9] Communication, S., Barrault, L., Chung, Y.-A., Cora Meglioli, M., Dale, D. et al. SeamlessM4T: Massively Multilingual & Multimodal Machine Translation. *ArXiv e-prints*. august 2023, p. arXiv:2308.11596. DOI: 10.48550/arXiv.2308.11596.

[10] Conneau, A. and Lample, G. Cross-lingual Language Model Pretraining. In: Wallach, H. M., Larochelle, H., Beygelzimer, A., Buc, F. d'Alché, Fox, E. A. et al., ed. *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*. 2019, p. 7057–7067. Available at: http://papers.nips.cc/paper/8928-cross-lingual-language-model-pretraining.

[11] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K. and Harshman, R. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*. 1990, vol. 41, no. 6, p. 391–407. DOI: https://doi.org/10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9. Available at: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-4571%28199009%2941%3A6%3C391%3A%3AAID-ASI1%3E3.0.CO%3B2-9.

[12] Duquenne, P.-A., Gong, H. and Schwenk, H. Multimodal and Multilingual Embeddings for Large-Scale Speech Mining. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P. and Vaughan, J. W., ed. *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2021, vol. 34, p. 15748–15761. Available at: https://proceedings.neurips.cc/paper_files/paper/2021/file/8466f9ace6a9acbe71f75762ffc890f1-Paper.pdf.

[13] Duquenne, P.-A., Schwenk, H. and Sagot, B. SONAR: Sentence-Level Multimodal and Language-Agnostic Representations. *ArXiv e-prints*. august 2023, p. arXiv:2308.11466. DOI: 10.48550/arXiv.2308.11466.

[14] Fan, J., Upadhye, S. and Worster, A. Understanding receiver operating characteristic (ROC) curves. *Canadian Journal of Emergency Medicine*. Cambridge University Press. 2006, vol. 8, no. 1, p. 19–20. DOI: 10.1017/S1481803500013336.

[15] Fapšo, M. *Query-by-Example Spoken Term Detection*. Brno, CZ, 2014. Disertační práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Available at: https://www.fit.vut.cz/study/phd-thesis/282/.

[16] Feng, F., Yang, Y., Cer, D., Arivazhagan, N. and Wang, W. Language-agnostic BERT Sentence Embedding. In: Muresan, S., Nakov, P. and Villavicencio, A., ed. *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, p. 878–891. DOI: 10.18653/v1/2022.acl-long.62. Available at: https://aclanthology.org/2022.acl-long.62.

[17] Golub, G. H. and Van Loan, C. F. *Matrix Computations*. 3rd ed. The Johns Hopkins University Press, 1996. 55 p. Johns Hopkins Studies in the Mathematical Sciences. ISBN 9780801854149.

[18] Guo, M., Shen, Q., Yang, Y., Ge, H., Cer, D. et al. Effective Parallel Corpus Mining using Bilingual Sentence Embeddings. In: Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y. et al., ed. *Proceedings of the Third Conference on Machine Translation: Research Papers*. Brussels, Belgium: Association for Computational Linguistics, October 2018, p. 165–176. DOI: 10.18653/v1/W18-6317. Available at: https://aclanthology.org/W18-6317.

[19] Harris, Z. S. Distributional Structure. *WORD*. Routledge. 1954, vol. 10, 2-3, p. 146–162. DOI: 10.1080/00437956.1954.11659520. Available at: https://doi.org/10.1080/00437956.1954.11659520.

[20] Hendrycks, D. and Gimpel, K. Gaussian Error Linear Units (GELUs). *ArXiv e-prints*. june 2016, p. arXiv:1606.08415. DOI: 10.48550/arXiv.1606.08415.

[21] Hsu, W.-N., Bolte, B., Tsai, Y.-H. H., Lakhotia, K., Salakhutdinov, R. et al. HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.* IEEE Press. oct 2021, vol. 29, p. 3451–3460. DOI: 10.1109/TASLP.2021.3122291. ISSN 2329-9290. Available at: https://doi.org/10.1109/TASLP.2021.3122291.

[22] Jang, E., Gu, S. S. and Poole, B. Categorical Reparameterization with Gumbel-Softmax. *ArXiv*. 2016, abs/1611.01144. Available at: https://api.semanticscholar.org/CorpusID:2428314.

[23] Jégou, H., Douze, M. and Schmid, C. Product Quantization for Nearest Neighbor Search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2011, vol. 33, no. 1, p. 117–128. DOI: 10.1109/TPAMI.2010.57.

[24] Kenton, J. D. M.-W. C. and Toutanova, L. K. Bert: Pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of naacL-HLT*. 2019, vol. 1, p. 2.

[25] Kürzinger, L., Winkelbauer, D., Li, L., Watzel, T. and Rigoll, G. CTC-Segmentation of Large Corpora for German End-to-End Speech Recognition. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2020, p. 267–278. DOI: 10.1007/978-3-030-60276-5_27. ISBN 9783030602765. Available at: http://dx.doi.org/10.1007/978-3-030-60276-5_27.

[26] Labrador, B., Zhao, G., Moreno, I. L., Scarpati, A. S., Fowl, L. H. et al. Exploring Sequence-to-Sequence Transformer-Transducer Models for Keyword Spotting. *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2022, p. 1–5. Available at: https://api.semanticscholar.org/CorpusID:253510109.

[27] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A. et al. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In: Jurafsky, D., Chai, J., Schluter, N. and Tetreault, J., ed. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, p. 7871–7880. DOI: 10.18653/v1/2020.acl-main.703. Available at: https://aclanthology.org/2020.acl-main.703.

[28] Martin, A., Doddington, G., Kamm, T., Ordowski, M. and Przybocki, M. A. The DET curve in assessment of detection task performance. In: *EUROSPEECH*. 1997.

[29] Mikolov, T., Chen, K., Corrado, G. and Dean, J. Efficient Estimation of Word Representations in Vector Space. *Proceedings of Workshop at ICLR*. january 2013, vol. 2013.

[30] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In: Burges, C., Bottou, L., Welling, M., Ghahramani, Z. and Weinberger, K., ed. *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2013, vol. 26. Available at: https://proceedings.neurips.cc/paper_files/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf.

[31] Muda, L., Begam, M. and Elamvazuthi, I. Voice Recognition Algorithms using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques. *J Comput*. march 2010, vol. 2, p. 138–143.

[32] NLLB Team, Costa-jussà, M. R., Cross, J., Çelebi, O., Elbayad, M. et al. No Language Left Behind: Scaling Human-Centered Machine Translation. *ArXiv e-prints*. july 2022, p. arXiv:2207.04672. DOI: 10.48550/arXiv.2207.04672.

[33] Nosratighods, M., Ambikairajah, E., Epps, J. and Carey, M. Score weighting in speaker verification systems. In: *2007 6th International Conference on Information, Communications Signal Processing*. 2007, p. 1–4. DOI: 10.1109/ICICS.2007.4449714.

[34] OSGOOD, C. E. Semantic Differential Technique in the Comparative Study of Cultures1. *American Anthropologist*. 1964, vol. 66, no. 3, p. 171–200. DOI: https://doi.org/10.1525/aa.1964.66.3.02a00880. Available at: https://anthrosource.onlinelibrary.wiley.com/doi/abs/10.1525/aa.1964.66.3.02a00880.

[35] Panayotov, V., Chen, G., Povey, D. and Khudanpur, S. Librispeech: An ASR corpus based on public domain audio books. In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2015, p. 5206–5210. DOI: 10.1109/ICASSP.2015.7178964.

[36] Pennington, J., Socher, R. and Manning, C. D. GloVe: Global Vectors for Word Representation. In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, p. 1532–1543. Available at: http://www.aclweb.org/anthology/D14-1162.

[37] Radford, A. and Narasimhan, K. Improving Language Understanding by Generative Pre-Training. In:. 2018. Available at: https://api.semanticscholar.org/CorpusID:49313245.

[38] Radford, A., Narasimhan, K., Salimans, T. and Sutskever, I. Improving language understanding with unsupervised learning. Technical report, OpenAI. 2018.

[39] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S. et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *CoRR*. 2019, abs/1910.10683. Available at: http://arxiv.org/abs/1910.10683.

[40] Rajaraman, A. and Ullman, J. D. *Mining of Massive Datasets*. Cambridge University Press, 2011. 1-17 p. ISBN 9781139505345.

[41] Safari, P., India, M. and Hernando, J. Self-attention encoding and pooling for speaker recognition. In: *Interspeech*. 2020. Available at: https://api.semanticscholar.org/CorpusID:220961537.

[42] Schneider, S., Baevski, A., Collobert, R. and Auli, M. Wav2vec: Unsupervised Pre-training for Speech Recognition. *CoRR*. 2019, abs/1904.05862. Available at: http://arxiv.org/abs/1904.05862.

[43] Schomacker, T. and Tropmann Frick, M. Language Representation Models: An Overview. *Entropy*. 2021, vol. 23, no. 11. DOI: 10.3390/e23111422. ISSN 1099-4300. Available at: https://www.mdpi.com/1099-4300/23/11/1422.

[44] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L. et al. Attention is all you need. *Advances in neural information processing systems*. Curran Associates, Inc. 2017, vol. 30. Available at: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[45] Velivelli, A., ChengXiang Zhai and Huang, T. S. Audio segment retrieval using a short duration example query. 2004, vol. 3, p. 1603–1606 Vol.3. DOI: 10.1109/ICME.2004.1394556.

[46] Vijayakumar, A., Vedantam, R. and Parikh, D. Sound-Word2Vec: Learning Word Representations Grounded in Sounds. In: Palmer, M., Hwa, R. and Riedel, S., ed. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, September 2017, p. 920–925. DOI: 10.18653/v1/D17-1096. Available at: https://aclanthology.org/D17-1096.

[47] Wu, J., Martin, A., Greenberg, C. S. and Kacker, R. The Impact of Data Dependence on Speaker Recognition Evaluation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*. 2017, vol. 25, p. 5–18.

[48] Yang, Y., Abrego, G., Yuan, S., Guo, M., Shen, Q. et al. Improving Multilingual Sentence Embedding using Bi-directional Dual Encoder with Additive Margin Softmax. In:. August 2019, p. 5370–5378. DOI: 10.24963/ijcai.2019/746.

[49] Yu, D., Gong, Y., Picheny, M. A., Ramabhadran, B., Hakkani Tür, D. et al. Twenty-Five Years of Evolution in Speech and Language Processing. *IEEE Signal Processing Magazine*. 2023, vol. 40, no. 5, p. 27–39. DOI: 10.1109/MSP.2023.3266155.

[50] Yujian, L. and Bo, L. A Normalized Levenshtein Distance Metric. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2007, vol. 29, no. 6, p. 1091–1095. DOI: 10.1109/TPAMI.2007.1078.

Appendix A

Results of the search algorithm

This appendix presents the detailed results obtained from the searches and the outputs of the command line.

A.1 LaBSE baseline

In this section, the outputs of the LaBSE model with the default settings of the search algorithm are presented. Both results for cosine distance (COS) and Euclidean distance (EUC) are provided. The search results for the keyword "start", without the enlarged window, thus $|win| = 1$:

```
<THRESH:0.5 COS (9, 10) 0.5969 0: I THOUGHT THAT WAS THE WAY TO [[BEGIN]]
       7850-281318-0015.txt
<THRESH:0.5 COS (13, 14) 0.5509 1: INDEED IT IS NOT A NEST AT ALL ONLY THE [[BEGINNING]] OF
     ONE
       7850-281318-0001.txt
<THRESH:0.5 COS (24, 25) 0.5485 2: ANYTHING WAS GOOD ENOUGH SO LONG AS IT PAID SAY FIVE
    DOLLARS A WEEK TO [[BEGIN]] WITH
       2277-149874-0016.txt
<THRESH:0.5 COS (20, 21) 0.5413 3: WHEN ARE YOU GETTING RID OF THESE CATS I'M NOT FIXING TO
     [[START]] AN ANNEX TO KATE'S CAT HOME
       1988-24833-0003.txt
<THRESH:0.5 COS (5, 6) 0.5410 4: WE HAD BETTER [[START]] THE DRIVE THIS MORNING
       6313-76958-0031.txt
<THRESH:0.5 COS (4, 5) 0.5331 5: AN EARLY [[START]] WAS MADE SO THAT THE PARTY REACHED THE
    PROMISED TABLE LANDS SHORTLY BEFORE TEN O'CLOCK IN THE FORENOON
       6313-66129-0018.txt
<THRESH:0.5 COS (1, 2) 0.5285 6: [[GEORGE]]
       3081-166546-0008.txt
<THRESH:0.5 COS (67, 68) 0.5265 7: IT IS OBVIOUS THAT EVERYWHERE THE DESIGNATIONS OF MORAL
    VALUE WERE AT FIRST APPLIED TO MEN AND WERE ONLY DERIVATIVELY AND AT A LATER PERIOD
    APPLIED TO ACTIONS IT IS A GROSS MISTAKE THEREFORE WHEN HISTORIANS OF MORALS [[START]]
    WITH QUESTIONS LIKE WHY HAVE SYMPATHETIC ACTIONS BEEN PRAISED
       422-122949-0003.txt

>THRESH:0.9 EUC (0, 0) 0.8718 0: [[GEORGE]]
       3081-166546-0008.txt
>THRESH:0.9 EUC (0, 0) 0.8750 1: [[YES]]
       3081-166546-0073.txt
>THRESH:0.9 EUC (9, 10) 0.8802 2: I THOUGHT THAT WAS THE WAY TO [[BEGIN]]
       7850-281318-0015.txt
>THRESH:0.9 EUC (3, 4) 0.8817 3: FESTIVE [[YES]]
```

```
       2428-83699-0001.txt
>THRESH:0.9 EUC (3, 4) 0.8854 4: [[LECTURES]]
       251-136532-0022.txt
>THRESH:0.9 EUC (3, 4) 0.8907 5: [[HONESTLY]]
       8297-275155-0021.txt
>THRESH:0.9 EUC (1, 2) 0.8915 6: AFTER [[THE]] VERY FIRST
       1462-170142-0024.txt
>THRESH:0.9 EUC (3, 4) 0.8948 7: THE [[TWENTIES]]
       1272-141231-0013.txt
>THRESH:0.9 EUC (0, 0) 0.8990 8: [[TEN]] SECONDS
       1272-141231-0016.txt
```

The search results for the keyword "start", with the enlarged window, thus $|win| = 1 + \lfloor(1/2 \times 1 + 1/2)\rfloor = 2$:

```
<THRESH:0.5 COS (8, 10) 0.5804 0: I THOUGHT THAT WAS THE [[WAY TO BEGIN]]
       7850-281318-0015.txt
<THRESH:0.5 COS (13, 15) 0.5428 1: INDEED IT IS NOT A NEST AT ALL ONLY [[THE BEGINNING]] OF
     ONE
       7850-281318-0001.txt
<THRESH:0.5 COS (23, 25) 0.5254 2: ANYTHING WAS GOOD ENOUGH SO LONG AS IT PAID SAY FIVE
    DOLLARS A WEEK [[TO BEGIN]] WITH
       2277-149874-0016.txt
<THRESH:0.5 COS (5, 7) 0.5179 3: WE HAD BETTER [[START]] THE DRIVE THIS MORNING
       6313-76958-0031.txt
<THRESH:0.5 COS (15, 17) 0.5177 4: IT SEEMED AS IF HIS FAMILY TROUBLES WERE [[JUST
    BEGINNING]]
       2277-149897-0024.txt
<THRESH:0.5 COS (19, 21) 0.5161 5: WHEN ARE YOU GETTING RID OF THESE CATS I'M NOT FIXING [[
    TO START]] AN ANNEX TO KATE'S CAT HOME
       1988-24833-0003.txt
<THRESH:0.5 COS (3, 5) 0.5154 6: THE [[TWENTIES]]
       1272-141231-0013.txt
<THRESH:0.5 COS (3, 5) 0.5135 7: AFTER THE [[VERY FIRST]]
       1462-170142-0024.txt
<THRESH:0.5 COS (1, 3) 0.5055 8: HE [[STARTED]] TO CONSCIOUS CONFUSION ONLY NEITHER KNOWING
     WHERE HE WAS NOR WHAT HE DID
       6295-64301-0027.txt
<THRESH:0.5 COS (18, 20) 0.5049 9: THAT TOOK THE CENTER OF INTEREST AWAY FROM ARCHAEOLOGY
    [[AND STARTED]] A NEW BURST OF ACTIVITY
       251-136532-0002.txt
<THRESH:0.5 COS (11, 13) 0.5047 10: THE BOYS WERE NOW ALL ANXIETY TO [[START]] WHILE THE
    PONIES AFTER THEIR SUNDAY REST WERE ALMOST AS FULL OF LIFE AS WERE THEIR OWNERS
       6313-66129-0016.txt
       2078-142845-0029.txt
<THRESH:0.5 COS (3, 5) 0.5042 11: THE SECOND [[PART BEGINS]] HERE I WAS A THINKING THE
    FIRST PART DIVIDES INTO TWO
       8842-302203-0008.txt
<THRESH:0.5 COS (3, 5) 0.5023 12: AN EARLY [[START]] WAS MADE SO THAT THE PARTY REACHED THE
     PROMISED TABLE LANDS SHORTLY BEFORE TEN O'CLOCK IN THE FORENOON
       6313-66129-0018.txt

>THRESH:0.81 EUC (8, 10) 0.7556 0: I THOUGHT THAT WAS THE [[WAY TO]] BEGIN
       7850-281318-0015.txt
>THRESH:0.81 EUC (13, 15) 0.7822 1: INDEED IT IS NOT A NEST AT ALL ONLY [[THE BEGINNING]]
    OF ONE
       7850-281318-0001.txt
>THRESH:0.81 EUC (5, 7) 0.7936 2: WE HAD BETTER [[START]] THE DRIVE THIS MORNING
       6313-76958-0031.txt
```

```
>THRESH:0.81 EUC (15, 17) 0.7963 3: IT SEEMED AS IF HIS FAMILY TROUBLES WERE [[JUST
    BEGINNING]]
       2277-149897-0024.txt
>THRESH:0.81 EUC (23, 25) 0.8027 4: ANYTHING WAS GOOD ENOUGH SO LONG AS IT PAID SAY FIVE
    DOLLARS A WEEK [[TO BEGIN]] WITH
       2277-149874-0016.txt
>THRESH:0.81 EUC (0, 1) 0.8028 5: [[TO MAKE]] HOT BUTTERED TOAST SEVENTEEN TWENTY SIX
       2078-142845-0029.txt
>THRESH:0.81 EUC (3, 5) 0.8057 6: THE SECOND [[PART BEGINS]] HERE I WAS A THINKING THE
    FIRST PART DIVIDES INTO TWO
       8842-302203-0008.txt
>THRESH:0.81 EUC (1, 3) 0.8085 7: HE [[STARTED]] TO CONSCIOUS CONFUSION ONLY NEITHER
    KNOWING WHERE HE WAS NOR WHAT HE DID
       6295-64301-0027.txt
```

Search results for the given query phrase "Nice day to meet" on Librispeech dev subset with the enlarged search window approach. First for cosine results and then for Euclidean distance:

```
Results for Nice day to meet:
<THRESH:0.5 COS (0, 7) 0.5543 0: [[TO MEET WAS TO FIND]] EACH OTHER
       174-168635-0009.txt
<THRESH:0.5 COS (7, 15) 0.5268 1: I HAD THE [[PLEASURE OF MEETING HIM IN]] SOCIETY
       3752-4944-0009.txt
<THRESH:0.5 COS (0, 7) 0.5248 2: [[HER MEETING WITH LETTY]] WAS INDESCRIBABLY TENDER AND
    THE DAYS THAT FOLLOWED WERE PRETTY EQUALLY DIVIDED BETWEEN HER AND HER BROTHER IN
    NURSING THE ONE AND LOVING THE OTHER
       3853-163249-0000.txt
<THRESH:0.5 COS (9, 17) 0.5182 3: WHICH THING BEING THUS THERE [[CAME A DAY WHEN CERTAIN]]
    LADIES TO WHOM IT WAS WELL KNOWN THEY HAVING BEEN WITH ME AT DIVERS TIMES IN MY TROUBLE
     WERE MET TOGETHER FOR THE PLEASURE OF GENTLE COMPANY
       8842-302201-0001.txt
<THRESH:0.5 COS (0, 7) 0.5181 4: [[YES I KNOW VERY]] WELL
       1462-170145-0020.txt
<THRESH:0.5 COS (2, 10) 0.5038 5: HE [[REALLY GRIEVED TO SEE IT]]
       6319-275224-0007.txt

>THRESH:0.63 EUC (0, 7) 0.6010 0: [[HER MEETING WITH LETTY]] WAS INDESCRIBABLY TENDER AND
    THE DAYS THAT FOLLOWED WERE PRETTY EQUALLY DIVIDED BETWEEN HER AND HER BROTHER IN
    NURSING THE ONE AND LOVING THE OTHER
       3853-163249-0000.txt
>THRESH:0.63 EUC (9, 17) 0.6043 1: WHICH THING BEING THUS THERE [[CAME A DAY WHEN CERTAIN]]
     LADIES TO WHOM IT WAS WELL KNOWN THEY HAVING BEEN WITH ME AT DIVERS TIMES IN MY
    TROUBLE WERE MET TOGETHER FOR THE PLEASURE OF GENTLE COMPANY
       8842-302201-0001.txt
>THRESH:0.63 EUC (0, 7) 0.6140 2: [[ON THE NEXT DAY BUT]] ONE RANDAL ARRANGED HIS DEPARTURE
     FOR SYDENHAM SO AS TO ARRIVE AT THE HOTEL AN HOUR BEFORE THE TIME APPOINTED FOR THE
    DINNER
       8297-275155-0000.txt
>THRESH:0.63 EUC (41, 49) 0.6204 3: WHETHER THEIR MANNER WAS GRAVE OR FRIVOLOUS HE KNEW
    THAT THESE WERE GOOD FRIENDS OF HIS AND HE SINCERELY HOPED THAT [[HE WOULD MEET THEM
    AGAIN]]
       6295-244435-0020.txt
>THRESH:0.63 EUC (0, 7) 0.6255 4: [[ONE DAY WHEN I]] RODE OVER TO THE SHIMERDAS I FOUND
    ANTONIA STARTING OFF ON FOOT FOR RUSSIAN PETER'S HOUSE TO BORROW A SPADE AMBROSCH
    NEEDED
       2035-147960-0002.txt
>THRESH:0.63 EUC (4, 12) 0.6258 5: ON THE [[ELEVENTH DAY WE SIGHTED]] CAPE PORTLAND OVER
    WHICH TOWERED MOUNT MYRDALS YOKUL WHICH THE WEATHER BEING CLEAR WE MADE OUT VERY
    READILY
```

```
       6241-61943-0008.txt
>THRESH:0.63 EUC (0, 7) 0.6285 6: [[THE DAY IS COMING]] SAID PROMETHEUS WHEN JUPITER WILL
    SEND A FLOOD TO DESTROY MANKIND FROM THE EARTH
       6319-57405-0005.txt
>THRESH:0.63 EUC (0, 7) 0.6288 7: [[THIS IS VERY GOOD]] OF YOU HE BEGAN GLANCING DOWN AT
    THE AGED DETECTIVE'S BUNDLED UP LEGS AND GENTLY PUSHING A CHAIR TOWARDS HIM
       3081-166546-0031.txt
>THRESH:0.63 EUC (2, 10) 0.6292 8: HE [[WOULD GET ONE TO DAY IT]] WOULD PROBABLY BE ON HIS
    DESK WHEN HE GOT BACK HE WOULD LOOK FOR IT AT ONCE
       2277-149896-0013.txt
```

A.2 SpeechT5 baseline

In this section, we present output results from the search algorithm. We present outputs obtained from the base and Hugging Face models with different collar settings. All of the following results are tested within `Librispeech dev-clean` dataset.

First, the base model output examples for the recorded query-by-example "place":

```
Results for ./eval/qbe/place.wav:
<THRESH:0.65 COS (65, 148) 0.7272 0: YOU'VE SOMETHING TO T[[ELL ME I SEE IT IN YOUR FACE]]
    DEAR I MUST GO
       3853-163249-0023.flac
<THRESH:0.65 COS (23, 106) 0.7216 1: THIS IS[[ THE PROBLEM OF RACE]]
       422-122949-0017.flac
<THRESH:0.65 COS (29, 112) 0.6861 2: ILLU[[STRATION RUSKS]]
       2078-142845-0049.flac
<THRESH:0.65 (76, 159) 0.6727 3: AT THE END OF IT SHE WAS [[IN A PLACE OF TOMBS]]
       6345-64257-0008.flac
```

For comparison, the same query-by-example processed by SpeechT5 Hugging Face model:

```
Results for ./eval/qbe/place.wav:
<THRESH:0.65 COS (185, 234) 0.7284 0: THEY SHOULD BE KEPT IN A CLOSED TIN CANISTER IN [[A
    DRY PLACE]] TO PRESERVE THEIR CRISPNESS
       2078-142845-0045.flac
<THRESH:0.65 COS (173, 222) 0.6752 1: A PERSON WOULD THINK THAT AFTER A FAMILY HAD LIVED SO
     LON[[G IN A PLACE ALL ]]THE NEIGHBORS WOULD BE FOND OF THEM YET IT IS NOT SO
       7850-286674-0000.flac
<THRESH:0.65 COS (305, 354) 0.6741 2: ...WE ALWAYS HAD LOGS OF WOOD BLAZING IN AN OPEN F[[
    IREPLACE AND]] SO DID MANY OTHER PEOPLE AND COAL...
       2803-161169-0010.flac
<THRESH:0.65 COS (11, 94) 0.6041 3: HE SHOWED HIM[[SELF ON THE PLATFORM]]
```

Example output obtained with the SpeechT5 Hugging Face model for the text input query "I had much pleasure in reading" for different collar sizes:

```
COLLAR=0.5
Results for I had much pleasure in reading:
<THRESH:0.35 COS (0, 45) 0.5835 0: [[I HAD THE PLEASURE OF MEETING HIM IN SOCIETY
]] 3752-4944-0009.txt
<THRESH:0.35 COS (5, 50) 0.5803 1: THEN [[I HAD MUCH PLEASURE IN READING IT BUT WAS IND]]
    EED SURPRISED AT THE MANY LITTLE POINTS...
       2412-153947-0003.txt
<THRESH:0.35 COS (0, 38) 0.4861 2: [[LEMON JUICE MAY BE ADDED AT PLEASURE]]
       1919-142785-0029.txt
<THRESH:0.35 COS (95, 140) 0.4299 3: ...HER HUSBAND'S EXERTIONS SHE[[ WOULD HAVE TAKEN
    PLEASURE IN READING EVERY W]]ORD OF THE EVIDENCE EVEN THOUGH HER HUSBAND...
```

```
       3536-8226-0005.txt
<THRESH:0.35 COS (3, 48) 0.4283 4: I E[[XPRESSED BY SIGNS MY ADMIRATION AND PLEASURE ]]TO
    MY GUIDES AND THEY WERE GREATLY PLEASED
       2412-153954-0011.txt
<THRESH:0.35 COS (65, 110) 0.4116 5: ...HIS MOTHER AND SISTER TOOK A [[PLEASURE IN
    CREDITING HER DAILY WITH SOME FRE]]SH AND UNPLEASING INSTRUMENT...
       6345-93306-0025.txt
<THRESH:0.35 COS (0, 44) 0.3972 6: [[AND AGAIN HE LISTENED WITH A QUIET PLEASURE
]] 6345-93306-0003.txt
<THRESH:0.35 COS (1, 46) 0.3918 7: H[[IS FACE HAD A LOOK OF WEARINESS AND PLEASURE ]]LIKE
    THAT OF SICK PEOPLE WHEN THEY FEEL RELIEF FROM PAIN
       1993-147965-0007.txt
<THRESH:0.35 COS (7, 52) 0.3777 8: MISTER [[POWER IS WAITING ARE YOU READY LOVE QUITE REA]]
    DY
       3853-163249-0050.txt
<THRESH:0.35 COS (29, 74) 0.3770 9: A MINUTE IS NOT A VERY LARGE [[MEASURE OF TIME AND HIS
    BODY NEEDED EVERY FRA]]CTION OF IT
       1272-141231-0005.txt

Collar=1.0
Results for I had much pleasure in reading:
<THRESH:0.30 COS (0, 46) 0.5812 0: [[I HAD THE PLEASURE OF MEETING HIM IN SOCIETY]]
       3752-4944-0009.txt
<THRESH:0.30 COS (7, 67) 0.5324 1: THEN I [[HAD MUCH PLEASURE IN READING IT BUT WAS INDEED
    SURPRISED AT ]]THE MANY LITTLE POINTS OF SIMILARITY...
       2412-153947-0003.txt
<THRESH:0.30 COS (0, 38) 0.4861 2: [[LEMON JUICE MAY BE ADDED AT PLEASURE]]
       1919-142785-0029.txt
<THRESH:0.30 COS (67, 127) 0.4212 3: ...HIS MOTHER AND SISTER TOOK A PL[[EASURE IN
    CREDITING HER DAILY WITH SOME FRESH AND UNPLEASING]] INSTRUMENT...
       6345-93306-0025.txt
<THRESH:0.30 COS (19, 79) 0.4200 4: A MINUTE IS NOT A V[[ERY LARGE MEASURE OF TIME AND HIS
    BODY NEEDED EVERY FRACTION]] OF IT
       1272-141231-0005.txt
<THRESH:0.30 COS (81, 141) 0.4152 5: ...THE JUDGE ORDINARY BY MEANS OF HER HUSBAND'S[[
    EXERTIONS SHE WOULD HAVE TAKEN PLEASURE IN READING EVERY WO]]RD OF THE...
       3536-8226-0005.txt
<THRESH:0.30 COS (0, 56) 0.3979 6: [[MISTER POWER IS WAITING ARE YOU READY LOVE QUITE READY
    ]]
       3853-163249-0050.txt
<THRESH:0.30 COS (0, 45) 0.3972 7: [[AND AGAIN HE LISTENED WITH A QUIET PLEASURE]]
       6345-93306-0003.txt
<THRESH:0.30 COS (0, 59) 0.3819 8: [[I EXPRESSED BY SIGNS MY ADMIRATION AND PLEASURE TO MY
    GUIDE]]S AND THEY WERE GREATLY PLEASED
       2412-153954-0011.txt
<THRESH:0.30 COS (7, 67) 0.3475 9: HIS FAC[[E HAD A LOOK OF WEARINESS AND PLEASURE LIKE
    THAT OF SICK PEO]]PLE WHEN THEY FEEL RELIEF FROM PAIN
       1993-147965-0007.txt

Collar=1.5
Results for I had much pleasure in reading:
<THRESH:0.35 COS (0, 46) 0.5812 0: [[I HAD THE PLEASURE OF MEETING HIM IN SOCIETY
]] 3752-4944-0009.txt
<THRESH:0.35 COS (3, 78) 0.5164 1: THE[[N I HAD MUCH PLEASURE IN READING IT BUT WAS INDEED
    SURPRISED AT THE MANY LI]]TTLE POINTS OF SIMILARITY...
       2412-153947-0003.txt
<THRESH:0.35 COS (0, 38) 0.4861 2: [[LEMON JUICE MAY BE ADDED AT PLEASURE]]
       1919-142785-0029.txt
<THRESH:0.35 COS (5, 80) 0.4341 3: A MIN[[UTE IS NOT A VERY LARGE MEASURE OF TIME AND HIS
    BODY NEEDED EVERY FRACTION ]]OF IT
```

```
       1272-141231-0005.txt
<THRESH:0.35 COS (53, 128) 0.4086 4: ...HIS MOTHER AND SI[[STER TOOK A PLEASURE IN
    CREDITING HER DAILY WITH SOME FRESH AND UNPLEASING ]]INSTRUMENT...
       6345-93306-0025.txt
<THRESH:0.35 COS (0, 56) 0.3979 5: [[MISTER POWER IS WAITING ARE YOU READY LOVE QUITE READY
    ]]
       3853-163249-0050.txt
<THRESH:0.35 COS (0, 45) 0.3972 6: [[AND AGAIN HE LISTENED WITH A QUIET PLEASURE]]
       6345-93306-0003.txt
<THRESH:0.35 COS (15, 90) 0.3964 7: I EXPRESSED BY [[SIGNS MY ADMIRATION AND PLEASURE TO MY
     GUIDES AND THEY WERE GREATLY PLEASED]]
       2412-153954-0011.txt
<THRESH:0.35 COS (59, 134) 0.3825 8: ...THE JUDGE ORDINARY BY [[MEANS OF HER HUSBAND'S
    EXERTIONS SHE WOULD HAVE TAKEN PLEASURE IN READING E]]VERY WORD...
       3536-8226-0005.txt

Collar=2.0
<THRESH:0.35 COS (0, 46) 0.5812 0: [[I HAD THE PLEASURE OF MEETING HIM IN SOCIETY]]
       3752-4944-0009.txt
<THRESH:0.35 COS (0, 38) 0.4861 1: [[LEMON JUICE MAY BE ADDED AT PLEASURE]]
       1919-142785-0029.txt
<THRESH:0.35 COS (5, 95) 0.4797 2: THEN [[I HAD MUCH PLEASURE IN READING IT BUT WAS INDEED
    SURPRISED AT THE MANY LITTLE POINTS OF SI]]MILARITY...
       2412-153947-0003.txt
<THRESH:0.35 COS (1, 91) 0.4262 3: I[[ EXPRESSED BY SIGNS MY ADMIRATION AND PLEASURE TO MY
    GUIDES AND THEY WERE GREATLY PLEASED]]
       2412-153954-0011.txt
<THRESH:0.35 COS (0, 87) 0.4236 4: [[A MINUTE IS NOT A VERY LARGE MEASURE OF TIME AND HIS
    BODY NEEDED EVERY FRACTION OF IT]]
       1272-141231-0005.txt
<THRESH:0.35 COS (0, 56) 0.3979 5: [[MISTER POWER IS WAITING ARE YOU READY LOVE QUITE READY
    ]]
       3853-163249-0050.txt
<THRESH:0.35 COS (0, 45) 0.3972 6: [[AND AGAIN HE LISTENED WITH A QUIET PLEASURE]]
       6345-93306-0003.txt
<THRESH:0.35 COS (49, 139) 0.3795 7: ...GIRL HIS MOTHER AN[[D SISTER TOOK A PLEASURE IN
    CREDITING HER DAILY WITH SOME FRESH AND UNPLEASING INSTRUMENT ]]COULD HAVE HAD...
       6345-93306-0025.txt
<THRESH:0.35 COS (43, 133) 0.3715 8: ...THE CASE BEEN BROUGHT BEFORE THE JU[[DGE ORDINARY
    BY MEANS OF HER HUSBAND'S EXERTIONS SHE WOULD HAVE TAKEN PLEASURE IN READING ]]EVERY...
       3536-8226-0005.txt
<THRESH:0.35 COS (0, 87) 0.3494 9: [[AT SUCH TIME ITS HEIGHT SEEMS MUCH LESS AS IF
    CROUCHING AND WEARY IT WERE TAKING REST]]
       3000-15664-0009.txt
```

A.3 SeamlessM4T based system

This appendix section presents examples of the SeamlessM4T model. These examples demonstrate the robustness of the SONAR embedding space and its quality.

The top results are shown for the phrase "It was the worst Sunday" entered as both QbT and QbE. The used dataset is Librispeech dev-clean.

```
Results for It was the worst Sunday:
COSINE >0.7:
<THRESH:0.7 COS (0, 21) 0.9125 0: [[IT WAS THE WORST SUNDAY HE HAD SPENT IN HI]]S LIFE
       2277-149897-0023.flac
<THRESH:0.7 COS (0, 22) 0.9109 1: [[IT WAS THE WORST SUNDAY HE HAD SPENT IN HIS ]]LIFE
       2277-149897-0023.txt
```

```
<THRESH:0.7 COS (0, 22) 0.8604 2: [[ON MONDAY THE TIDE WAS]] REVERSED
       5694-64025-0013.flac
<THRESH:0.7 COS (0, 23) 0.8399 3: [[THE ACCIDENT IN QUESTION OCCURRED UPON THE SUNDAY]]
    EVENING
       2428-83705-0006.flac
<THRESH:0.7 COS (0, 29) 0.8196 4: [[HE SAW A BUSY SATURDAY USHERED OUT THE SABBATH IN ]]AND
     NOTHING DONE
       2277-149897-0021.flac
<THRESH:0.7 COS (0, 14) 0.8096 5: [[IT WAS A HORRIBLE JOUR]]NEY
       2428-83699-0014.flac
<THRESH:0.7 COS (0, 14) 0.8092 6: [[IT WAS A HORRIBLE JOURNEY]]
       2428-83699-0014.txt
<THRESH:0.7 COS (0, 25) 0.7891 7: [[BUT THE WOOD PIGEON WAS IN THE WORST CASE OF THEM ]]ALL
       7850-281318-0022.flac
<THRESH:0.7 COS (0, 30) 0.7589 8: [[ABOUT DAYLIGHT ON SUNDAY MORNING CHALMERS BRIGADE
    RELIEVED G]]LADDEN'S
       5694-64025-0003.flac
<THRESH:0.7 COS (0, 29) 0.7488 9: THEY SAT ABOUT [[THE HOUSE MOST OF THE DAY AS IF IT WERE
    SUNDAY GREASING]] THEIR BOOTS MENDING THEIR SUSPENDERS PLAITING WHIPLASHES
       1993-147964-0000.flac

       Results for eval/qbe/it was the worst sunday.wav:
COSINE >0.7:
<THRESH:0.7 COS (0, 22) 0.9204 0: [[IT WAS THE WORST SUNDAY HE HAD SPENT IN HIS ]]LIFE
       2277-149897-0023.txt
<THRESH:0.7 COS (0, 21) 0.9200 1: [[IT WAS THE WORST SUNDAY HE HAD SPENT IN HI]]S LIFE
       2277-149897-0023.flac
<THRESH:0.7 COS (0, 18) 0.8882 2: [[ON MONDAY THE TIDE]] WAS REVERSED
       5694-64025-0013.txt
<THRESH:0.7 COS (0, 14) 0.8579 3: [[IT WAS A HORRIBLE]] JOURNEY
       2428-83699-0014.txt
<THRESH:0.7 COS (0, 24) 0.8376 4: [[BUT THE WOOD PIGEON WAS IN THE WORST CASE OF THE]]M ALL
       7850-281318-0022.flac
<THRESH:0.7 COS (0, 24) 0.8274 5: [[IT WAS THE AFTERNOON OF A HOLIDAY AND SHE HAD CL]]OSED
    EARLY
       6295-64301-0001.txt
<THRESH:0.7 COS (0, 14) 0.8273 6: [[IT WAS A HORRIBLE JO]]URNEY
       2428-83699-0014.flac
<THRESH:0.7 COS (0, 23) 0.7870 7: [[IT WAS ONE WHICH GAVE ME A SMALL TRIUMPH OVER ]]GEORGE
       3081-166546-0028.txt
<THRESH:0.7 COS (0, 24) 0.7269 8: [[HE SAW A BUSY SATURDAY USHERED OUT THE SABBATH I]]N AND
     NOTHING DONE
       2277-149897-0021.flac
<THRESH:0.7 COS (0, 24) 0.7167 9: [[IT WAS THE AFTERNOON OF ]]A HOLIDAY AND SHE HAD CLOSED
    EARLY
       6295-64301-0001.flac
```